Vol. 32, No. 1, 2023

Machine GRAPHICS & VISION

International Journal

Published by
The Institute of Information Technology
Warsaw University of Life Sciences – SGGW
Nowoursynowska 159, 02-776 Warsaw, Poland
in cooperation with
The Association for Image Processing, Poland – TPO

Vision-Based Biomechanical Markerless Motion Classification

Yu Liang Liew, Jeng Feng Chin
School of Mechanical Engineering, Universiti Sains Malaysia, 14300 Nibong Tebal, Penang, Malaysia
chinjengfeng@usm.my

Abstract. This study used stick model augmentation on single-camera motion video to create a markerless motion classification model of manual operations. All videos were augmented with a stick model composed of keypoints and lines by using the programming model, which later incorporated the COCO dataset, OpenCV and OpenPose modules to estimate the coordinates and body joints. The stick model data included the initial velocity, cumulative velocity, and acceleration for each body joint. The extracted motion vector data were normalized using three different techniques, and the resulting datasets were subjected to eight classifiers. The experiment involved four distinct motion sequences performed by eight participants. The random forest classifier performed the best in terms of accuracy in recorded data classification in its min-max normalized dataset. This classifier also obtained a score of 81.80% for the dataset before random subsampling and a score of 92.37% for the resampled dataset. Meanwhile, the random subsampling method dramatically improved classification accuracy by removing noise data and replacing them with replicated instances to balance the class. This research advances methodological and applied knowledge on the capture and classification of human motion using a single camera view.

Key words: vision, single camera, markerless, stick model, human motion, motion classification, data mining.

1. Introduction

Human motion analysis entails sensing the human body and extracting static or dynamic data from it in the form of gestures, behaviors, and actions [34]. It emerges as a critical component in operation studies to evaluate performance, such as in sports performance analysis [16], medical rehabilitation [61], video surveillance [18], and virtual reality gaming [28]. In industrial engineering, motion classification aids in verifying the presence of operator action, and the absence of specific actions can lead to process defects and incompletion [1], as well as safety concerns [22].

Fixed-axis and parallel projection are used in vision-based motion classification models to calibrate feature points relative to the previous position of human body parts [53]. The general framework of a vision-based motion classification model includes movement scene capture, human tracking, humans and motion representation, motion recognition, and classification into its respective class [37]. In general, the model processes each frame of the motion video in accordance with its frame sequences. When a human is detected in a video frame, the frame image is segmented to obtain the region of interest [41]. The motion can then be visualized by combining a stick-figure model, a volumetric model, 2D

blobs, and a geometric drawing [2]. Among these methods, the stick-figure model provides a simple but effective solution for estimating a human posture at a specific frame. The stick-figure model is a skeleton-like model composed of several keypoints, each of which represents a coordinate of a body part. These body parts function as moving joints, and their motion vectors are compared with those of the previous frame [10]. The motion is classified by comparing the movement of the person between frames [52].

Most motion capture methods place markers on the body parts of the subject to track the change in motion. However, such a setting necessitates a planned experiment environment with informed subjects, which makes it impractical in a real-life scenario where preparation or interference with the observed activity is not permitted. Several studies used multi-camera recording to reconstruct the 3D view of moving human bodies in the absence of motion capture markers. Nakano et al. [38] used multiple video cameras from different angles to capture frames from various perspectives, which they then merged into 3D visualization using the direct linear transformation method. Meanwhile, Hasler et al. [23] used audio synchronization to conventional video camera recordings and then 3D mesh reconstruction using a feature-based approach.

Kanko et al. [26] used a single 2D camera view to perform gait analysis and movement estimation using a deep learning approach. Tsuji et al. [49] used a single camera to capture video of general movements of infants and identify abnormalities in those movements. Then, they utilized a framework that begins with feature extraction using computer vision and progresses to movement analysis using formula calculations. Finally, they conducted movement classification using a feedforward-type network known as a log-linearized Gaussian mixture network. Zult et al. [63] demonstrated that a conventional video camera could extract the valid keypoints of body parts in the video frame based on the markers using a low-cost 2D camera system. Using a computer vision module such as OpenPose [6], the markers could be replaced by virtual coordinate points [57]. For example, Kim et al. [27] used the OpenPose module to predict knee and hip movement angles in a video captured with a smartphone camera. The validity of this OpenPose-based system with the automated post-processing algorithm has shown early promise, but it may require further verification.

This study investigated the markerless motion classification approach, with motion video captured using a single 2D camera view. The markerless motion classification model classified manual operations extracted from motion video by using the stick model augmentation. This study has two objectives. The first objective is to develop a descriptive model for motion classification based on the overlay of a stick-figure model onto the motion of the operator in video frames. The second objective is to determine the best motion classification strategy by assessing the accuracy of the motion classification model using data mining classifier algorithms. The research advances methodological and applied knowledge on the capture and classification of human motion using a single

camera view. The use of a single camera has cost, configuration, and maintenance benefits. The method can be used in real-world industry applications, such as in analyzing operator performance during a repetitive manufacturing process.

The structure of the manuscript is provided. It begins with an overview of the research context, followed by a brief review of the literature on human motion segmentation, stick-figure models, and motion classification. The following section 3 describes the research methodology, which includes the experiment setup, motion data extraction and computation, and motion classification. Section 4 contains the results and discussion. The final section 5 elaborates the conclusion.

2. Literature review

Motion segmentation is a preprocessing stage of motion analysis that is used to cluster long frame sequences depicting human actions into several shorter, non-overlapping video segments. Subspace clustering and temporal data clustering are two popular clustering methods in the literature. Subspace clustering works by searching a dataset for subspaces and clusters and categorizing data into new distinct spaces based on similar features. For example, Xia et al. [55] combined sparse subspace clustering and a robust kernel low-rank representation method for motion recognition. However, the method ignores the temporal correlation between successive frames. Temporal data clustering divides large amounts of sequential data into non-overlapping chunks. Wang et al. [51] highlighted the importance of temporal information in achieving accurate model performance. However, transfer learning is required to overcome the unpredictability of the results because the temporal clustering method is unsupervised.

Several recent studies used transfer learning to visualize object motion using existing datasets, which is due to that prior knowledge from related source data improves feature identification. Several works partially, such as [62] or fully adapted transfer learning by using deep neural network classifier parameters. They are useful, particularly for detecting multiple people in the same image frame [45].

Rubino et al. [42] proposed semantic motion detection, which uses semantic information to identify object matches between two views. Its underlying principle is similar to that of the convolutional neural network model, which uses patterns from training data to identify features in target data. Simonyan and Zisserman [46] proposed a two-stream convolutional network model with spatial and temporal networks. With prior knowledge of training data from the optical flow model, this model identifies the moving action in the testing video. Meanwhile, Zhou and He [60] used the recurrent network model to estimate the human body region in the image by transforming the image into a pose heatmap. The heatmap would be used to evaluate the coordinates of body joints, and these coordinates are critical for building the stick-figure model.

A stick-figure model is a skeleton-like structure used to represent important body

joints and track body motion patterns [21]. Annotations of keypoints from the body pose estimation are used to accomplish this model. Handcrafted features, such as histogram of oriented gradient, are used in the previous stick model. However, the accuracy of the identified keypoints is below the acceptable range [13]. Single-person or multi-body human body position estimations are used in modern times. The single-person approaches locate body parts through direct regression and heatmap conversion [14]. Chan et al. [9] used a mathematical regression coefficient model to simplify the 2D stick model of human motion for direct regression. Figure 1 depicts the construction of the 2D stick model, which is composed of several points of body parts and lines. The model presents a more straightforward interpretation by using joints as calculation points.

However, the regression-based stick model construction always necessitates additional procedures to accurately map the feature points onto the subject in an image. Carreira et al. [8] added a corrective measure to the neural network model structure by including a simple error feedback connection. The predicted error was fed back into the network in

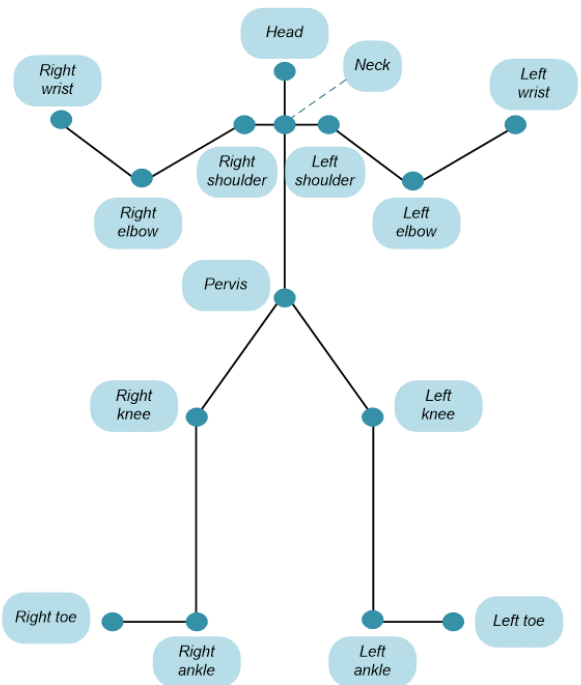


Fig. 1. A simple two-dimensional stick model.

the form of backpropagation to gradually improve keypoint location prediction. Luvizon et al. [35] presented the soft-argmax operation, which is an improved method. After this operation is integrated into the deep convolutional neural network, it can convert the feature maps directly to joint coordinates by finding the maximum values from the target functions. This new method produces results that are comparable to those of the heatmap-based framework. However, unlike the heatmap-based approach, expanding this method into multi-person cases is problematic.

The detection-based framework is typically built on deep learning datasets that have been pre-trained using thousands of human images. Sun et al. [47] used a convolutional neural network with two-stride convolutions to reduce resolution and a main body that outputs feature maps to implement their approach. At the network's end, the regressor estimates the keypoint positions by evaluating the loss function of the heatmap using comparisons between predicted and ground-truth heatmaps.

Various motion classification techniques have been proposed. Switonski et al. [48] investigated data mining for markerless motion extraction in motion capture data. They used dynamic time warping (DTW) technique to classify the human motion data into gait patterns. In time-series data, the model identifies variations in the orientation of motion capture and subject for motion recognition. It calculates the angles in the joint data and the classification probability with the minimum distance classifiers (MDC). MDC is combined with k-nearest neighbor classifiers to maximize the accuracy and consistency of both types of classifiers. Schneider et al. [44] used the DTW approach to evaluate warping distance after annotating the skeleton model using the OpenPose module dataset. Prior to applying the classifier, the image data in coordinates were normalized to condense the data range into a smaller number. Thereafter, nearest neighbor classifiers were used to classify the warping distance of time-series data. The results still have some limitations, such as reliance on the representativeness of the dataset, poor recognition precision when noise reduction is required, and the need for motion capture marker setup.

Qian et al. [40] evaluated multi-class support vector machine (SVM) classifiers by removing the background and extracting the centroids and instantaneous speed of human motion. The frame sequence comparison produces a contour coding of motion energy image with a square-to-circular coordinate transformation, which converts plane coordinates to polar coordinates. SVMs were also used as classifiers in the study by Choi et al. [11] study to classify the gait motion pattern. The joint angle and distances between body parts are among the parameters used. SVM is an excellent option for accurately recognizing motion, but many more classifiers have yet to be tested in motion classification.

Yang and Zhao [58] used decision tree classifiers to determine the motion class of firefighters, but string-type descriptions rather than numbers were utilized as attributes. Zhang et al. [59] employed an interactive system to classify six different motions using three classifiers: naïve Bayes, SVM, and random forest. The results showed that the

Tab. 1. Descriptions for experimental motion activities.

Motion Activity	Description
Moving box	Bend down the body, lift the box with two hands, stand upright, walk a few steps, bend down the body, put down the box, resume to a standing position.
Moving pail	Bend down the body, lift pail by its handle with one hand, stand upright, walk a few steps, bend down the body, put down the pail, resume to a standing position.
Sweeping	Grasp a broom with one hand, move the broom down until its brush touching the floor, pull the broom to sweep the dirt, lift the broom up.
Mopping	Bend down the body, lift the box with two hands, stand upright, walk a few steps, bend down the body, place the box down, and resume standing. Bend down the body, lift the pail by the handle with one hand, stand upright, walk a few steps, bend down the body, set the pail down, and resume standing. Grasp a broom with one hand, lower the broom until the brush touches the floor, pull the broom to sweep the dirt, and then raise the broom. Grasp a mop with two hands, slightly bend the body, move the mop in one direction until it touches the floor, then reverse the mop movement.

random forest classifier has the highest classification accuracy when using position and vector data. Li et al. [31] investigated the motion recognition model using the random forest algorithm and the difference in normalized joint coordinates between keyframes. Fong et al. [17] agreed that the random forest classifier performs the best using position and vector data from the skeleton model. It outperforms the neural network approach and other traditional classifiers in terms of classification accuracy.

3. Methodology

3.1. Experimental motion selection

The experiment was designed to involve activities observable in common full body operations. As described in Table 1, the motion activities featured in the experiment are moving carton box, moving pail, sweeping floor, and mopping floor. Moving carton box and moving pail are highly similar operations, as well as sweeping and mopping floor. The intention is to create complexity in learning when the system is being presented with highly similar datasets.

The variation in human action influences pose recognition. Eight participants between the ages of 23 and 24 volunteered for the motion video collection to account for the abovementioned effect. Each participant was required to complete a series of aforementioned activities in various settings. Different backgrounds (outdoor and indoor) and light conditions were used in the video sample collection given that video backgrounds affected motion recognition using a markerless system [5]. The outdoor used natural light, whereas the indoor light conditions could be bright or dim.

The motion classification samples were collected at the university student hostels. The motion recording was conducted with a digital single-lens reflex (DSLR) camera, specifically a Nikon DSLR D3200 model, with a frame rate of 60 frames per second and a video frame size of 7201080 pixels in three color channels. During video capture, a camera tripod stand supports the camera and fixes its position. Figure 2 depicts and labels the camera setup parallel to the motion activity. During video capture, each participant was required to face the camera parallelly.

A participant repeated each motion activity 10 times, which were recorded all in the same video. All sample videos were manually trimmed into individual activity videos by using video editing software. The first three segmented videos of each sample video were considered motion warm-up and were excluded from the subsequent processing stage. A total of 100 videos from each motion class were chosen at random for further processing. All 400 videos were uploaded to Google Drive in folders named after the motion class to be processed by programming.

The stick model augmentation estimates body part position using the COCO dataset [33]. By associating joint coordinates with individuals, the COCO dataset has been used in multi-person tracking and keypoint annotations [30]. The COCO dataset contains over 200,000 labeled object instances and at least 250,000 human samples. The dataset includes annotations and information for all body part instances, which aid in segmentation and estimation of keypoint coordinates. The COCO dataset was used to train the model for object detection and estimation using transfer learning. As shown in Table 2, 18 points per person had to be recognized from the COCO dataset onto each human image. Python was used to annotate the stick model keypoints and lines onto the human body in the video frames for the stick model overlay. The body joint position was estimated using OpenPose [24], which has been integrated with OpenCV [36].

With 4D blobs, the image was converted into image data. The blobs were fed into the trained neural network, which identified the maximum points in the object area and detected the objects. The architecture of the pre-trained network was divided into two branches: the top branch, which predicts the confidence map, and the bottom branch, which estimates the affinity field. Affinity field refers to the storage of unstructured pairwise relationships between body parts in a field [7].

The code was written in Python and executed in Google Colab [20] using the Python 3

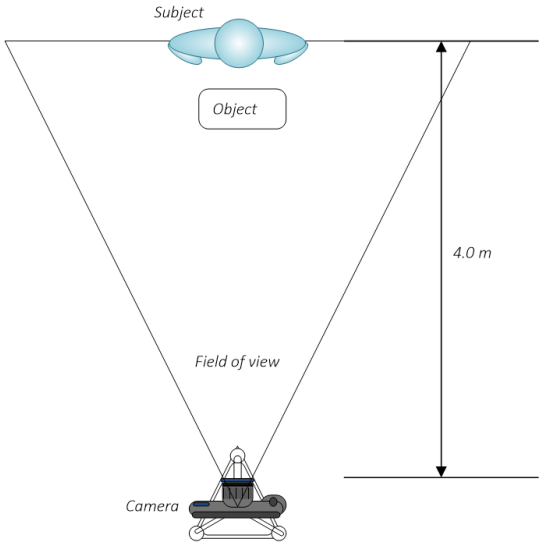


Fig. 2. Motion video capturing scene setup.

Tab. 2. Representation of each number for body joints.

Number	Body Joint	Number	Body Joint
0	Nose	9	Right knee
1	Neck	10	Right ankle
2	Right shoulder	11	Left hip
3	Right elbow	12	Left knee
4	Right wrist	13	Left ankle
5	Left shoulder	14	Right eye
6	Left elbow	15	Left eye
7	Left wrist	16	Right ear
8	Right hip	17	Left ear

Google Compute Engine Tensor processing unit backend [19] with 35.25 GB of high-RAM. All videos were saved in Google Drive folders. The COCO dataset was imported, and the keypoints were sequentially paired (Table 3) with a different number to represent the various body joints identified in Table 2.

The flowchart (Figure 3) summarized the algorithm for overlaying the stick figure. To save computation power, all experimental motion videos were annotated with keypoints and line connections based on specified body part pairings every half a second.

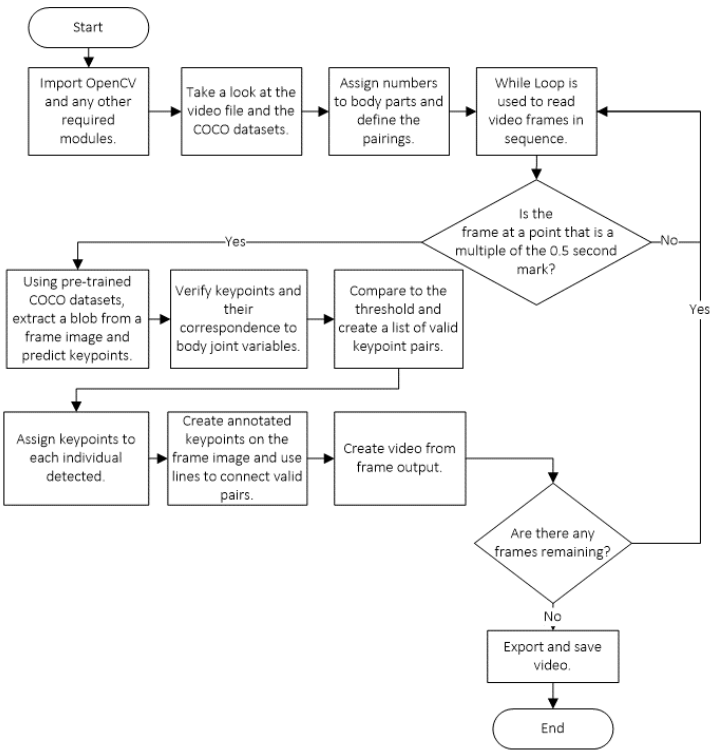


Fig. 3. Flowchart for programming model augmentation using stick figures.

3.2. Data collection and calculation for motion

Keypoint estimation in the stick model overlay was used to calculate the position coordinates for each body joint in a frame. Positions, velocity, and acceleration were among the data extracted from the stick-figure model. Eyes and ears were irrelevant to motion evaluation. Thus, only the first 14 body joints were counted in the extracted data. Owing to the single 2D view of the video, the initial velocities of body parts were calculated for the x- and y-axes only in the motion classification model. Meanwhile, the cumulative velocity and acceleration were used to account for the time-series effect of the video. Table 4 lists the extracted motion variables with n, which indicates the representation number of body joint plus one. The representation number of body joint can be found in Table 2. The initial velocities of body joints were calculated using Equations (1) and (2) for x- and y axes, respectively.

$$u_{x_n} = \frac{x_{1_n} - x_{0_n}}{t}, \tag{1}$$

$$u_{y_n} = \frac{y_{1_n} - y_{0_n}}{t}, \tag{2}$$

where:
x_{1_n} – x-axis coordinate at the first instance for body joint n,
x_{0_n} – x-axis coordinate at the start for body joint n,
y_{1_n} – y-axis coordinate at the first instance for body joint n,
y_{0_n} – y-axis coordinate at the start for body joint n,
t – time interval, here equal to 0.5.

Equations (3) and (4) were used to calculate the cumulative velocity of a body part in the x and y directions, respectively. Meanwhile, (5) and (6) defined the equations for calculating the cumulative acceleration of body parts in the x and y-axes, respectively.

Tab. 3. Body joints pairing with the number indication.

Number Pair	Body Joints Pairing	Number Pair	Body Joints Pairing
1,2	Neck – Right shoulder	11,12	Left hip – Left knee
1,5	Neck – Left shoulder	12,13	Left knee – Left ankle
2,3	Right shoulder – Right elbow	1,0	Neck – Nose
3,4	Right elbow – Right wrist	0,14	Nose – Right eye
5,6	Left shoulder – Left elbow	14,16	Right eye – Right ear
6,7	Left elbow – Left wrist	0,15	Nose – Left eye
1,8	Neck – Right hip	15,17	Left eye – Left ear
8,9	Right hip – Right knee	2,17	Right shoulder – Left ear
9,10	Right knee – Right ankle	5,16	Left shoulder – Right ear
1,11	Neck – Left hip		

Tab. 4. Initial variables and vector variables for motion data extraction.

Initial Velocity Variables	Description	Vector Variables	Description
u_{x_n}	Initial velocity of nth body part at x-axis.	v_{x_n}	Cumulative velocity in the x-direction of nth body part.
u_{y_n}	Initial velocity of nth body part at y-axis.	v_{y_n}	Cumulative velocity in the y-direction of nth body part.
		a_{x_n}	Cumulative acceleration in the x-direction of nth body part.
		a_{y_n}	Cumulative acceleration in the y-direction of nth body part.

$$v_{x_n} = \sum_{i=1}^{m} \frac{x_{i_n} - x_{(i-1)_n}}{t_i - t_{i-1}} \, t_i \,, \tag{3}$$

$$v_{y_n} = \sum_{i=1}^{m} \frac{y_{i_n} - y_{(i-1)_n}}{t_i - t_{i-1}} \, t_i \,, \tag{4}$$

$$a_{x_n} = \sum_{i=1}^{m} \frac{x_{i_n} - x_{(i-1)_n}}{(t_i - t_{i-1})^2} \, t_i \,, \tag{5}$$

$$a_{y_n} = \sum_{i=1}^{m} \frac{y_{i_n} - y_{(i-1)_n}}{(t_i - t_{i-1})^2} \, t_i \,, \tag{6}$$

where:
i – the instance index,
m – total number of frames in the video divided by 30,
t – time interval, here equal to 0.5,
n – body joint number (0 to 13) + 1,
x_{i_n} – x-axis coordinate at i^{th} instance for body joint n,
$x_{(i-1)_n}$ – x-axis coordinate at the previous instance for body joint n,
y_{i_n} – y-axis coordinate at i^{th} instance for body joint n,
$y_{(i-1)_n}$ – y-axis coordinate at the previous instance for body joint n.

These formulas were then incorporated into the programming algorithm. Each position and vector variable had 14 attributes to represent different body joints. Thus, the total number of attributes was 84, and a motion type class attribute was added at the end. All attributes were extracted and saved in a comma-separated values (CSV) file for use in data preprocessing and mining.

Several issues contribute to value errors from the annotation of the stick model, and they would be addressed differently. One of the issues in estimating coordinates is the inability to detect body parts due to a blocked view. Motion videos feature human subjects interacting with objects to perform the required activity. As a result, the interacted object is likely to become an impediment to viewing body parts from the motion video. An example is undetected feet by the programming algorithm due to the carton box obscuring its view. Aside from being blocked by objects, some body parts for keypoint detection are kept out of camera view by the pother body parts of the participant. The most notable occurrences involve sweeping and mopping, in which some participants choose to turn their bodies in different directions while performing the action. In both cases, missing coordinate data were replaced with estimated vector data. The COCO keypoint dataset uses a large amount of image data to detect human body part positions and estimate missing keypoints by comparison with other body parts [32]. This estimation method is valid only for common gestures like parallel standing and

lifting objects. The reason is that the dataset only has a few images for each pose. The issues were resolved by assuming that body part movement momentum continued from the previous frame to the current frame. The position of an undetected body part was estimated using the coordinates of the previous frame plus the instantaneous velocity of that body part.

Another error is mistaking unrelated objects for body part keypoints. These objects are identified as human body parts by Setjo et al. [45]. Using the multi-person dataset for keypoint estimation, multiple sets of keypoints are detected. However, separating humans from false positives is required. Thus, the bottom-up approach [6] of associating joints to people was used to reduce misidentification.

3.3. Data preprocessing

Data preprocessing steps are critical for preparing data for an effective data mining process. Several techniques were used to normalize the data extracted from the stick model. Then, the outliers and extreme values of normalized data were calculated before reacting. Duplicate instances were also identified in the preprocessing stage.

Motion data are normalized to standardize the range of different units or scales in the attributes. It simplifies large-number numeric attributes and improves data quality without affecting the final data classification result [25]. Three popular normalization techniques were used in this study: min-max normalization (MMN), Z-score normalization (ZSN), and decimal scaling normalization (DSN). MMN reduces the un-normalized data to a specific lower and upper boundary, which is typically 0 to 1 or -1 to 1. The formula in Equation (7) was used to calculate MMN [43].

$$v'_{i,n} = \frac{v_{i,n} - \min(v_n)}{\max(v_n) - \min(v_n)}(\max_{\text{new}} - \min_{\text{new}}) + \min_{\text{new}}\,, \tag{7}$$

where:
$v'_{i,n}$ – new normalized variable data at i^{th} instance,
$v_{i,n}$ – original variable data at i^{th} instance,
$\min(v_n)$ – minimum value of variable data in the n^{th} attribute,
$\max(v_n)$ – maximum value of variable data in the n^{th} attribute,
$\min_{\text{new}}$ – new minimum value, usually -1 or 0,
$\max_{\text{new}}$ – new maximum value, usually 1.

The ZSN method uses mean and standard deviation to normalize data into a scaled value ranging from -1 to 1, with zero mean and unit variance. ZSN is expressed by (8).

$$v'_{i,n} = \frac{v_{i,n} - \mu_n}{\sigma_n}\,, \tag{8}$$

where:
$v'_{i,n}$ – new normalized variable data at i^{th} instance,
$v_{i,n}$ – original variable data at i^{th} instance,
μ_n – mean of all data in the n^{th} attribute,
σ_n – standard deviation of all data in the n^{th} attribute.

DSN measures the maximum values of an attribute and rescales them by moving the decimal point of instance values. This method of normalization is useful for data with logarithmic variation in the attribute. In (9), the DSN formula is written as follows.

$$v'_{i,n} = \frac{v_{i,n}}{10^j} \tag{9}$$

where:
$v'_{i,n}$ – new normalized variable data at i^{th} instance,
$v_{i,n}$ – original variable data at i^{th} instance,
$j = \log_{10}(\max(v_n))$.

Google Colab was loaded with the CSV file containing the extracted motion data from the stick-figure model. The three normalization methods were applied using the Python Scikit-learn (Sklearn) module [12], which resulted in three different normalized CSV dataset files.

The normalized datasets were then resampled in the WEKA interface [54] (version 3.8.5) under the supervised instance filter section using a random subsampling method. Its goal was to improve the instances by removing noise from the motion data. The imbalanced result from different subsets created during cross-validation could be due to noise instances. With or without replacement, the random subsampling method generated a random subsample of a dataset. The data were balanced with replicated instances from the remaining data to maintain the same class bias as the original unprocessed dataset without compromising the total sampling number for the experiment. The three datasets were preprocessed and saved to new CSV files before the motion classification experiment was started.

3.4. Motion classification

The WEKA Experimenter was used to run the motion classification experiment, which included all three normalized datasets and eight different classifiers (Table 5). The default WEKA settings were used except for the options in brackets that required manual input. Each classifier was run 10 times. The 10-fold cross-validation option was used to divide the training and validation data into 10 sets, with each set serving as the testing set iteratively in 10 rounds of validation. For each data preprocessing technique, the experiment was repeated with resampled datasets. A total of 4800 experimental trials were conducted.

Tab. 5. Classifiers used in the experiment.

Classifier	Description
ZeroR	The most basic rule-based classifiers predict the majority class while ignoring all predictors or attributes [15].
OneR	Selects the single most informative attribute and classifies instances solely on the basis of this attribute's criteria [39].
J48 Decision Tree (pruned)	Produces pruned trees that begin at the root node and classify instances into branches by sorting them according to attribute values [29].
Random forest	Building many individual decision trees with each random forest tree results in a class prediction, and the class with the most votes becomes the final model's prediction.
Random tree	The decision tree and Random Forest approaches are combined to predict the class by fitting several decision tree classifiers on different sub-samples of the dataset and averaging to improve prediction accuracy and avoid over-fitting.
k-Nearest neighbors ($k = 5$)	A lazy learner method that classifies instances based on evaluated Euclidean distances that define the closeness to each class, where k represents the number of neighbours considered to find the majority of a class label [29].
Naïve Bayes	Calculates the conditional probability of the classes based on the assumption that each attribute is independent of the others [56].
Multilayer perceptron	It is made up of neural network layers, which include input, hidden, and output layers. Back-propagation is used to train neurons to process data and recognize patterns [50].

4. Results and discussion

4.1. Stick model overlay

The stick model was used to annotate all 400 videos, and body part keypoints and lines indicated the connection between body joints. Figure 4 depicts video frames with human body parts augmented by the stick model when moving a carton box, moving a pail, sweeping, and mopping, in that order.

A set of 100 videos from the same motion class took an average of 27 min to complete the stick model overlay process. A motion video contains 8 to 10 frames that are designated for stick model processing and data extraction. As a result, a single frame took between 1.62 and 2.03 s to complete the stick model augmentation process.

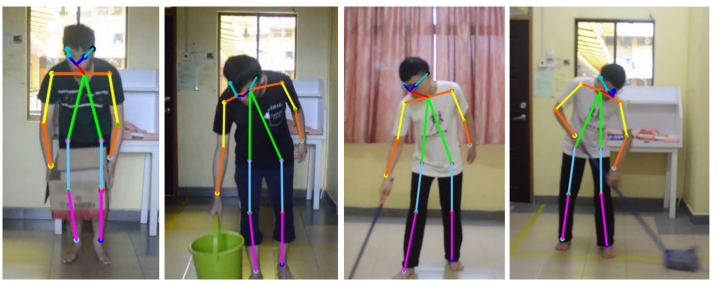


Fig. 4. Sample video frame with stick model overlay.

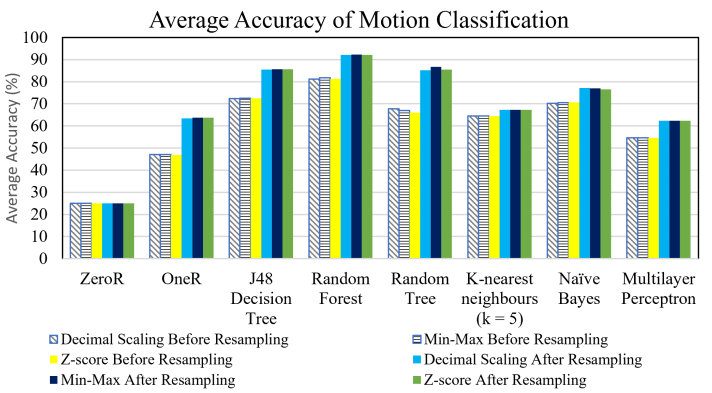


Fig. 5. Graph of average accuracy for all motion classification experimental trials.

4.2. Motion classification

A total of 4800 data mining experimental trials were conducted, which involved variable permutations of eight classifiers, three normalization techniques, and the use of resampling prior to classification. The average accuracies for each classifier and normalization technique permutation, with or without resampling, were evaluated. They are recorded in Table 6 and plotted as a graph in Figure 5.

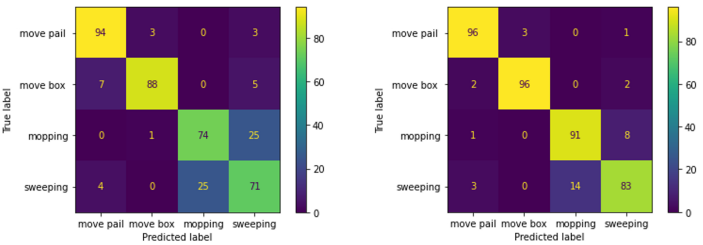
Except for the ZeroR classifier, the average classification accuracy for the datasets after resampling is higher than that for the datasets before resampling. ZeroR classifier maintains an accuracy of 25% regardless of normalization methods or resampling. The reason is that the ZeroR classifier frequently identifies the majority class. However, the majority class does not exist in these datasets because of the motion data distribution. As a result, the accuracy for all normalized datasets with a ZeroR classifier is the same.

Tab. 6. Classification accuracy of different classifiers and normalization technique used before and after the resampling.

Classifier	Normalization Technique	Average Accuracy without Resampling (%)	Average Accuracy with Resampling (%)
ZeroR	Decimal scaling	25.00	25.00
	Min-Max	25.00	25.00
	Z-score	25.00	25.00
OneR	Decimal scaling	47.15	63.45
	Min-Max	47.13	63.70
	Z-score	46.97	63.70
J48 Decision Tree (pruned)	Decimal scaling	72.38	85.52
	Min-Max	72.47	85.62
	Z-score	72.57	85.65
Random forest	Decimal scaling	81.25	92.10
	Min-Max	81.80	92.37
	Z-score	81.40	92.15
Random tree	Decimal scaling	67.78	85.20
	Min-Max	67.00	86.80
	Z-score	66.08	85.55
k-Nearest neighbors ($k = 5$)	Decimal scaling	64.53	67.30
	Min-Max	64.53	67.30
	Z-score	64.53	67.30
Naïve Bayes	Decimal scaling	70.30	77.20
	Min-Max	70.52	77.05
	Z-score	70.62	76.55
Multilayer perceptron	Decimal scaling	54.58	62.32
	Min-Max	54.58	62.32
	Z-score	54.58	62.32

Figure 5 shows that the random forest classifier method achieves the highest accuracy in the datasets before and after resampling categories. The random forest classifier with MMN and resampling has the best performance of the knowledge discovery method combination, with an average accuracy of 92.37%. The finding echoes previous studies on movement or gait analysis [17, 59]. The random forest classifier avoids overfitting in large datasets like the motion dataset. The motion dataset has 84 attributes, which can easily cause overfitting using other classifiers. The normalization techniques produce insignificant differences in classification accuracy while using the same classifiers. Thus, the normalization scale difference insignificantly affects the classification result. Nevertheless, the random forest classifier performs best with the min-max normalized dataset.

The resampling method increases classification accuracy by removing noise or misclassified data and replicating the remaining data to fill the void [4]. Confusion matrices



(a) without resampling (b) with resampling

Fig. 6. Confusion matrix for classification result of the min-max normalized dataset using Random Forest classifier.

for classification results of datasets with and without resampling confirm this explanation. As shown in Figure 6a, the random forest classifier and MMN produce a confusing confusion matrix for the dataset. Figure 6b shows the classification result of the same data mining technique for resampled data, which has a higher accuracy.

According to confusion matrices, sweeping and mopping are more likely to be misclassified due to their high similarity. The experimental motion capture does not impede movement execution. It affects the classification accuracy, especially for motions with similar characteristics. Resampling increases correctly classified instances in mopping and sweeping. The resampling method replaces incorrectly classified instances with replicated instances from correctly classified instances. Arbelaitz et al. [3] agreed that random subsampling improves accuracy. However, they recommended using synthetic minority oversampling technique (SMOTE) to obtain significant statistical differences between class instances. Future research should examine the effect of the SMOTE technique on the dataset.

5. Conclusion

This study develops a descriptive model for markerless motion classification using a single camera view. The stick model overlay uses OpenCV and OpenPose modules as well as COCO datasets. In motion classification, the best data mining strategy is determined by classifier and normalization accuracy. The best classifier is the random forest classifier, which achieves an accuracy of 81%-82% without resampling and an accuracy of 92%-93% with resampling. Using the same classifier, normalization techniques have little to no effect on classification accuracy. The developed algorithm of stick-figure model augmentation and data mining strategy complete the markerless motion classification model. This study can be extended to more complex and variable motion activities in manufacturing, such as manual operations.

Acknowledgement

This work was supported by The Malaysia Ministry of Higher Education (Kementerian Pengajian Tinggi) under Fundamental Research Grant Scheme (FRGS) grant no: FRGS/1/2021/TK0/USM/02/25.

References

[1] M. Aehnelt, E. Gutzeit, and B. Urban. Using activity recognition for the tracking of assembly processes: Challenges and requirements. In *Workshop on Sensor-Based Activity Recognition (WOAR)*, page 12–21, Rostock, Germany, Mar 2014. `https://publica.fraunhofer.de/entities/publication/45148487-5ac9-49c4-8ae8-e07e33aa87ef/details`.

[2] J. K. Aggarwal and Q. Cai. Human motion analysis: A review. *Computer Vision and Image Understanding*, 73(3):428–440, 1999. doi:10.1006/cviu.1998.0744.

[3] O. Arbelaitz, I. Gurrutxaga, J. Muguerza, and J. M. Pérez. Applying resampling methods for imbalanced datasets to not so imbalanced datasets. In *Conference of the Spanish Association for Artificial Intelligence*, page 111–120, 2013. doi:10.1007/978-3-642-40643-0_12.

[4] R. De Bin, S. Janitzaa, W. Sauerbrei, and A. L. Boulesteix. Subsampling versus bootstrapping in resampling-based model selection for multivariable regression. *Biometrics*, 72(1):272–280, 2016. doi:10.1111/biom.12381.

[5] M. Bosch, F. Zhu, and E. J. Delp. Video coding using motion classification. In *15th IEEE International Conference on Image Processing*, page 1588–1591, 2008. doi:10.1109/ICIP.2008.4712073.

[6] Z. Cao, G. Hidalgo, T. Simon, et al. OpenPose: Realtime multi-person 2D pose estimation using part affinity fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43(1):172–186, 2021. doi:10.1109/TPAMI.2019.2929257.

[7] Z. Cao, T. Simon, S. E. Wei, and Y. Sheikh. Realtime multi-person 2D pose estimation using part affinity fields. In *Proc. 30th IEEE Conference on Computer Vision and Pattern Recognition*, page 1302–1310, Jan 2017. doi:10.1109/CVPR.2017.143.

[8] J. Carreira, P. Agrawal, K. Fragkiadaki, and J. Malik. Human pose estimation with iterative error feedback. In *Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, page 4733–4742, Dec 2016. doi:10.1109/CVPR.2016.512.

[9] C. K. Chan, W. P. Loh, and I. A. Rahim. Human motion classification using 2d stick-model matching regression coefficients. *Applied Mathematics and Computation*, 283:70–89, 2016. doi:10.1016/j.amc.2016.02.032.

[10] M. G. Choi, K. Yang, T. Igarashi, et al. Retrieval and visualization of human motion data via stick figures. *Computer Graphics Forum*, 31(7):2057–2065, 2012. doi:10.1111/j.1467-8659.2012.03198.x.

[11] W. Choi, L. Li, H. Sekiguchi, and K. Hachimura. Recognition of gait motion by using data mining. In *International Conference on Control, Automation and Systems*, page 1213–1216, 2013. doi:10.1109/ICCAS.2013.6704173.

[12] D. Cournapeau, M. Brucher, F. Pedregosa, et al. scikit-learn. Machine Learning in Python, 2023. `https://scikit-learn.org`. [Accessed 15 Jan 2022].

[13] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *Proc. 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 1, page 886–893, 2005. doi:10.1109/CVPR.2005.177.

[14] Q. Dang, J. Yin, B. Wang, and W. Zheng. Deep learning based 2D human pose estimation: A survey. *Tsinghua Science and Technology*, 24(6):663–676, 2019. doi:10.26599/TST.2018.9010100.

[15] L. Devasena C. Effectiveness analysis of ZeroR, RIDOR and PART classifiers for credit risk appraisal. *International Journal of Advances in Computer Science and Technology (IJACST)*, 3(11):6–11, 2014. Special issue of ICCAAC 2014. `https://www.warse.org/IJACST/static/pdf/file/iccaac2014sp02.pdf`.

[16] R. Ferdinands. Advanced applications of motion analysis in sports biomechanics. In *Proc. XXVIII International Symposium of Biomechanics in Sports*, page 70–73, Jul 2010. `https://ojs.ub.uni-konstanz.de/cpa/article/view/4383`.

[17] S. Fong, J. Liang, I. Fister, and S. Mohammed. Gesture recognition from data streams of human motion sensor using accelerated PSO swarm search feature selection algorithm. *Journal of Sensors*, 2015:205707, 2015. doi:10.1155/2015/205707.

[18] G. B. Garibotto. 3-D computer vision modeling in video surveillance applications. In C. H. Chen, editor, *Handbook of Pattern Recognition and Computer Vision*, page 747–765. World Scientific, 2009. doi:10.1142/9789814273398_0033.

[19] Google. Cloud Tensor Processing Units (TPUs), 2022. `https://cloud.google.com/tpu`. [Accessed 15 Jan 2022].

[20] Google. Colaboratory, 2022. `https://colab.research.google.com`. [Accessed 15 Jan 2022].

[21] Y. Guo, G. Xu, and S. Tsuji. Tracking human body motion based on a stick figure model. *Journal of Visual Communication and Image Representation*, 5(1):1–9, 1994. doi:10.1006/jvci.1994.1001.

[22] S. U. Han, S. H. Lee, and F. Peña-Mora. Vision-based motion detection for safety behavior analysis in construction. In *Construction Research Congress 2012: Construction Challenges in a Flat World, Proc. 2012 Construction Research Congress*, page 1032–1041, 2012. doi:10.1061/9780784412329.104.

[23] N. Hasler, B. Rosenhahn, T. Thormahlen, et al. Markerless motion capture with unsynchronized moving cameras. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, page 224–231, 2009. doi:10.1109/CVPR.2009.5206859.

[24] G. Hidalgo, Z. Cao, T. Simon, et al. OpenPose, 2022. `https://github.com/CMU-Perceptual-Computing-Lab/openpose`. [Accessed 15 Jan 2022].

[25] A. Jahan and K. L. Edwards. A state-of-the-art survey on the influence of normalization techniques in ranking: Improving the materials selection process in engineering design. *Materials and Design*, 65(1):335–342, 2015. doi:10.1016/j.matdes.2014.09.022.

[26] R. M. Kanko, E. K. Laende, G. Strutzenberger, et al. Assessment of spatiotemporal gait parameters using a deep learning algorithm-based markerless motion capture system. *Journal of Biomechanics*, 122(110414), 2021. doi:10.1016/j.jbiomech.2021.110414.

[27] J. S. Kim, Y. W. Kim, Y. K. Woo, and K. N. Park. Validity of an artificial intelligence-assisted motion-analysis system using a smartphone for evaluating weight-bearing activities in individuals with patellofemoral pain syndrome. *Journal of Musculoskeletal Science and Technology*, 5(1):34–40, 2021. doi:10.29273/jmst.2021.5.1.34.

[28] S. Kloiber, V. Settgast, C. Schinko, et al. Immersive analysis of user motion in VR applications. *Visual Computer*, 36(10-12):1937–1949, 2020. doi:10.1007/s00371-020-01942-1.

[29] S. W. Knox. *Survey of Classification Techniques*. Wiley Series in Probability and Statistics, 2018. doi:10.1002/9781119439868.ch4.

[30] N. Le, A. Heili, and J. Odobez. Long-term time-sensitive costs for CRF-based tracking by detection.

In *European Conference on Computer Vision Workshops, Lecture Notes in Computer Science*, volume 9914, pages 43–51, 2016. doi:10.1007/978-3-319-48881-3_4.

[31] B. Li, B. Bai, and C. Han. Upper body motion recognition based on key frame and random forest regression. *Multimedia Tools and Applications*, 79(7-8):5197–5212, 2020. doi:10.1007/s11042-018-6357-y.

[32] T. Y. Lin, M. Maire, S. Belongie, et al. Research on face recognition based on CNN. In *Microsoft COCO: Common objects in context*, volume 8693 of *Lecture Notes in Computer Science*, page 740–755, 2014. doi:10.1007/978-3-319-10602-1_48.

[33] T.-Y. Lin, G. Patterson, M. R. Ronchi, et al. COCO. Common Objects in Context, 2020. `https://cocodataset.org`. [Accessed 6 Jan 2022].

[34] H. Liu, Z. Ju, X. Ji, C. S. Chan, and M. Khoury. *Human Motion Sensing and Recognition*. Springer, Berlin Heidelberg, 2017. `https://link.springer.com/book/10.1007/978-3-662-53692-6`.

[35] D. C. Luvizon, H. Tabia, and D. Picard. Human pose regression by combining indirect part detection and contextual information. *Computers and Graphics*, 85:15–22, 2019. doi:10.1016/j.cag.2019.09.002.

[36] OpenCV Team. OpenCV, 2022. `https://opencv.org`. [Accessed 15 Jan 2022].

[37] A. N. Mohamed and M. M. Ali. Human motion analysis, recognition and understanding in computer vision: A review. *Journal of Engineering Sciences*, 41(5):1928–1946, 2013. doi:10.21608/jesaun.2013.114925.

[38] N. Nakano, T. Sakura, K. Ueda, et al. Evaluation of 3D markerless motion capture accuracy using OpenPose with multiple video cameras. *Frontiers in Sports and Active Living*, 2(50):1–9, 2020. doi:10.3389/fspor.2020.00050.

[39] C. G. Nevill-Manning, G. Holmes, and I. H. Witten. The development of Holte's 1R classifier. In *Proc. 1995 2nd New Zealand International Two-Stream Conference on Artificial Neural Networks and Expert Systems*, page 239–242, Jan 1995. doi:10.1109/ANNES.1995.499480.

[40] H. Qian, Y. Mao, W. Xiang, and Z. Wang. Recognition of human activities using SVM multi-class classifier. *Pattern Recognition Letters*, 31(2):100–111, 2010. doi:10.1016/j.patrec.2009.09.019.

[41] J. Rittscher and A. Blake. Classification of human body motion. In *Proc. IEEE International Conference on Computer Vision*, volume 1, page 634–639, 1999. doi:10.1109/iccv.1999.791284.

[42] C. Rubino, M. Crocco, V. Murino, and A. Del Bue. Semantic multi-body motion segmentation. In *Proc. 2015 IEEE Winter Conference on Applications of Computer Vision, WACV 2015*, page 1145–1152, 2015. doi:10.1109/WACV.2015.157.

[43] C. Saranya and G. Manikandan. A study on normalization techniques for privacy preserving data mining. *International Journal of Engineering and Technology*, 5(3):2701–2704, 2013. `http://www.enggjournals.com/ijet/docs/IJET13-05-03-273.pdf`.

[44] P. Schneider, R. Memmesheimer, I. Kramer, and D. Paulus. Gesture recognition in RGB videos using human body keypoints and dynamic time warping. In *Lecture Notes in Computer Science*, volume 11531, page 281–293, 2019. doi:10.1007/978-3-030-35699-6_22.

[45] C. H. Setjo, B. Achmad, and Faridah. Thermal image human detection using Haar-cascade classifier. In *Proc. 2017 7th International Annual Engineering Seminar*, pages 1–6, 2017. doi:10.1109/INAES.2017.8068554.

[46] K. Simonyan and A. Zisserman. Two-stream convolutional networks for action recognition in videos. In *Proc. 27th International Conference on Neural Information Processing Systems*, volume 27 of *NIPS Proceedings*, page 568–576, 2014. `https://ora.ox.ac.uk/objects/uuid:1dd0bcd0-39ca-48a1-9c20-5341d6c49251`.

[47] K. Sun, B. Xiao, D. Liu, and J. Wang. Deep high-resolution representation learning for human pose estimation. In *Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, page 5686–5696, June 2019. doi:10.1109/CVPR.2019.00584.

[48] A. Switonski, H. Josinski, and K. Wojciechowski. Dynamic time warping in classification and selection of motion capture data. *Multidimensional Systems and Signal Processing*, 30(3):1437–1468, 2019. doi:10.1007/s11045-018-0611-3.

[49] T. Tsuji, S. Nakashima, H. Hayashi, et al. Markerless measurement and evaluation of general movements in infants. *Scientific Reports*, 10(1):1–13, 2020. doi:10.1038/s41598-020-57580-z.

[50] J. Wang and Z. Li. Research on face recognition based on CNN. In *IOP Conference Series: Earth and Environmental Science*, volume 170, page 032110, 2018. doi:10.1088/1755-1315/170/3/032110.

[51] L. Wang, Z. Ding, and Y. Fu. Low-rank transfer human motion segmentation. *IEEE Transactions on Image Processing*, 28(2):1023–1034, 2019. doi:10.1109/TIP.2018.2870945.

[52] L. Wang, W. Hu, and T. Tan. Recent developments in human motion analysis. *Pattern Recognition*, 36(3):585–601, 2003. doi:10.1016/S0031-3203(02)00100-0.

[53] J. A. Webb and J. K. Aggarwal. Visually interpreting the motion of objects in space. *Computer*, 14(8):40–46, 1981. doi:10.1109/C-M.1981.220561.

[54] I. H. Witten, E. Frank, M. A. Hall, and C. J. Pal. *The WEKA Workbench. Online Appendix for "Data Mining: Practical Machine Learning Tools and Techniques"*. Morgan Kaufmann, 2016. [Accessed 21 Sep 2021]. `https://www.cs.waikato.ac.nz/ml/weka/book.html`.

[55] G. Xia, H. Sun, L. Feng, et al. Human motion segmentation via robust kernel sparse subspace clustering. *IEEE Transactions on Image Processing*, 27(1):135–150, 2018. doi:10.1109/TIP.2017.2738562.

[56] X. Xie, J. W. K. Ho, C. Murphy, et al. Testing and validating machine learning classifiers by metamorphic testing. *Journal of Systems and Software*, 84(4):544–558, 2011. doi:10.1016/j.jss.2010.11.920.

[57] Q. Xu, G. Huang, M. Yu, and Y. Guo. Fall prediction based on key points of human bones. *Physica A: Statistical Mechanics and Its Applications*, 540:123205, 2020. doi:10.1016/j.physa.2019.123205.

[58] L. Yang and T. Zhao. Data mining and ergonomic evaluation of firefighter's motion based on decision tree classification model. In *Advanced Research on Computer Science and Information Engineering: International Conference: Proceedings*, volume Part 2, page 212–217, 2011. doi:10.1007/978-3-642-21411-0_35.

[59] H. Zhang, W. Du, and H. Li. *Kinect Gesture Recognition for Interactive System*. Stanford University Term Paper for CS 299, 2012. `https://cs229.stanford.edu/proj2012/ZhangDuLi-KinectGestureRecognitionforInteractiveSystem.pdf`.

[60] D. Zhou and Q. He. PoSeg: Pose-aware refinement network for human instance segmentation. *IEEE Access*, 8:15007–15016, 2020. doi:10.1109/aCCESS.2020.2967147.

[61] H. Zhou and H. Hu. Human motion tracking for rehabilitation—A survey. *Biomedical Signal Processing and Control*, 3(1):1–18, 2008. doi:10.1016/j.bspc.2007.09.001.

[62] T. Zhou, H. Fu, C. Gong, et al. Multi-mutual consistency induced transfer subspace learning for human motion segmentation. In *Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, page 10274–10283, 2020. doi:10.1109/CVPR42600.2020.01029.

[63] T. Zult, J. Allsop, J. Tabernero, and S. Pardhan. A low-cost 2-D video system can accurately and reliably assess adaptive gait kinematics in healthy and low vision subjects. *Scientific Reports*, 9(1):1–11, 2019. doi:10.1038/s41598-019-54913-5.

Yu Liang Liew currently works as a mechanical design engineer since 2021. He received his Bachelor of Engineering in Manufacturing Engineering with Management from the School of Mechanical Engineering at Universiti Sains Malaysia (USM). His interests include computer vision, artificial intelligence, big data and statistical analysis.

Jeng Feng Chin received his Ph.D. degree in Manufacturing Engineering from The University of Birmingham, United Kingdom. He has been an associate professor in the School of Mechanical Engineering at Universiti Sains Malaysia (USM) in Penang, Malaysia, since 2006. His research interests comprise computer-integrated manufacturing, lean manufacturing, production management, machine learning, artificial intelligence and optimization.

Exploring Automated Object Detection Methods for Manholes using Classical Computer Vision and Deep Learning for Autonomous Vehicles

Shika Rao[1], Nitya Mitnala[1,2]
[1]*Birla Institute of Technology and Science, Pilani – Hyderabad Campus, India*
[2]*Texas Instruments (India): Bengaluru, Karnataka, India*

Abstract. Open, broken, and improperly closed manholes can pose problems for autonomous vehicles and thus need to be included in obstacle avoidance and lane-changing algorithms. In this work, we propose and compare multiple approaches for manhole localization and classification like classical computer vision, convolutional neural networks like YOLOv3 and YOLOv3-Tiny, and vision transformers like YOLOS and ViT. These are analyzed for speed, computational complexity, and accuracy in order to determine the model that can be used with autonomous vehicles. In addition, we propose a size detection pipeline using classical computer vision to determine the size of the hole in an improperly closed manhole with respect to the manhole itself. The evaluation of the data showed that convolutional neural networks are currently better for this task, but vision transformers seem promising.

Key words: computer vision, object detection, size detection, Convolutional Neural Networks, Vision Transformers, autonomous vehicles

1. Introduction

The development of manholes is an ongoing trend spurred by urban growth. Though the construction of manholes is such an integral part of road development in the country, we see many incidents where the improper construction and maintenance of manholes have led to many road accidents, even resulting in the loss of lives. In India, at least two people die every day due to open manholes, according to [22]. These accidents occur due to broken manhole lids, improperly placed lids, or disproportional manhole lids which do not cover the manhole entirely. This problem is especially worsened during heavy rains wherein waterlogged roads can hinder even human driving.

Autonomous vehicles are not a reality yet in developing countries like India for many reasons as seen in [5, 27]. However, even seemingly trivial problems to human-driving could pose major obstacles to autonomous vehicles. One such problem that has not been accounted for in prior research is the existence of open or broken manholes which can lead to costly damages to the vehicle and pose a hindrance to public safety. Potholes are also a major problem, however the cost of damage to the vehicle would not be as high as that for open manholes. Thus, the focus of our research is manholes. Autonomous vehicles will soon be a reality even in developing countries [27] so assuming that the roads will have no problems with manholes could be catastrophic. Therefore, the inclusion of detection of open/broken manholes is imperative in obstacle avoidance and lane-changing

algorithms for self-driving vehicles. Humans can identify open manholes easily, however for autonomous vehicles this devolves into a computer vision problem.

In this paper, we focus on manhole detection and classification using computer vision. Autonomous vehicles have a camera and GPS for path planning and localization respectively. Using a camera, open or broken manholes can be localized and classified within an image, and using a GPS, the location of the manhole can be recorded to send to the concerned authorities. We collected a dataset of road-surface images and a few aerial images from Google Street View, Google Images, and [39]. A lot of the images in the dataset are from [39] and these images were collected using a moving vehicle and a GoPro HERO6 Black RGB camera with resolutions of 1280×720 pixels. Since the road-surface images were collected from a camera attached to a moving vehicle, it makes it very close to what an autonomous vehicle would encounter. The images include varied lighting conditions and can hence be used for real-time detection and classification of manholes on roads by self-driving vehicles.

The training images of the deep learning vision models we have trained, can also be used for detection on video streams. We also take this research one step ahead to further classify open and broken (improperly closed) manholes into high and low importance for determining the priority order of fixing the manhole. This can be used by private organizations or government authorities to easily find the faulty manholes, thus preventing unfortunate accidents.

In the past, image segmentation and classification based approaches have proven useful for crack detection [28] and pothole detection [21]. Considering this, and the fact that autonomous vehicles have cameras for driving, we focus on an object detection approach to manhole classification and localization. In this paper we evaluate and compare the different object detection models on the same dataset based on the computational complexity. The first approach we tested was classical computer vision for localization of the manhole in the images followed by a simple neural network for classification. This method is favorable for autonomous vehicles as it does not require a GPU or much computational power. In addition, our dataset was minimal with only 1032 images, thus we expected this method to be advantageous. However, it was not close to the expected benchmark, so we attempted simultaneous object localization and classification (object detection) using Convolutional Neural Networks (CNNs). Specifically, we first attempted YOLOv3-Tiny [1] which we hypothesized would give decent results without a heavy dependence on computational power. To compare the performance and the tradeoff between time and computational complexity, we applied YOLOv3 [37]. Vision Transformers (ViTs) [9] have gained a lot of popularity in object detection tasks in recent research (see [7]), thus we used YOLOS [12]. The results were similar to the classical computer vision approach, thus we implemented a simple, non-hierarchical vision transformer for just classification without localization, to evaluate its accuracy especially after

recent research reported in [24] highlighting its promising applications as a backbone for object detection tasks.

In this paper, we go a step further than just object detection for localizing and classifying manholes. We propose a classical computer vision pipeline to measure the size of the hole in a broken manhole with respect to the manhole itself to analyze the level of damage done to the manhole. We take the images classified as open and improperly closed using the object detection model and crop them around the bounding box drawn by the object detection model. We then filter and find the maximum contour in the image and compare it with the smaller contour found. If any of the smaller contours are greater than 50% of the total manhole area, we determine that the vehicle should stop or definitely avoid the manhole. This is proposed because although broken and open manholes pose a risk of damage to autonomous vehicles, all of them cannot be avoided, especially when the traffic is heavy and speed of the vehicle is high. This pipeline acts as a form of direction to classify the level of damage into high and low importance, which would be useful for obstacle avoidance in autonomous vehicles, and it could even be useful for the respective authorities to determine which manholes are in need of immediate attention to prevent major accidents. As autonomous vehicles are a reality these days (though a minority), automation in reporting the improper manholes to the authorities in charge using the vehicles' camera and GPS itself is possible.

Thus, the main contributions and novelty of this research work are as follows:

1. In prior research work on autonomous vehicles, the problems that manholes pose, especially in developing countries, are not taken into account. Our work highlights this problem and proposes solutions for it. Open manholes are a tangible risk to public safety, hence, we focus on improperly closed and open manholes in this paper.
2. We conduct a thorough literature review on previous work in manhole detection in road surface images. We also include a theoretical review of object detection algorithms.
3. We propose and evaluate multiple approaches to manhole localization and classification considering the trade-off between computational complexity and accuracy.
4. We propose a novel pipeline for elliptical object localization, classification, and bounding box prediction using classical computer vision.
5. In prior art, only Convolutional Neural Networks (CNNs) have been considered till now for the purpose of manhole detection. We test Vision Transformers specifically for manhole detection in self-driving vehicles.
6. Using classical computer vision techniques, we propose a pipeline to determine the size of the hole in a broken manhole with respect to the manhole image itself (size detection).
7. The results of our research show that Vision Transformers are promising as backbones for object detection, however currently Convolutional Neural Networks can be integrated in obstacle avoidance techniques of autonomous vehicles.

The rest of this manuscript is structured as follows: Section 2 reviews the state-of-the-art research about manhole detection using computer vision approaches. This section discusses the primary research gaps identified and describes the role of our research. Our decisions regarding the approaches chosen for the methodology are also elaborated upon in this section. Section 3 investigates the methodology of the various object detection methods we attempted for manhole object detection and size detection. Section 4 discusses the results of the paper and summarizes the outcomes of the training and testing process. The results are analyzed based on computational complexity, accuracy, and speed. Section 5 reiterates the main results of this work and concludes the manuscript by identifying the future avenues of work.

2. Review of literature

Research on manhole covers has been done over the years, and the methods of research are ever-changing with the emergence of new technologies. The commonality between existing and upcoming research on manhole cover detection is that all the methods are based on digital imagery. Though traditionally, broken and open manhole covers are detected and fixed through manual surveys and crowd reporting, this method is laborious, time-consuming, and ill-planned. For this reason, there is a demand for methods of automation like classical computer vision and deep learning. The papers are organized by the date of publication to understand the flow of research methodologies.

2.1. Classical computer vision approaches for manhole detection in road-surface images

A morphological method (dependent on the structure) was developed in [42] (2000) for detecting round-shaped manhole covers. It involved a *black top-hat transform* for feature extraction designed with disc-shaped structuring elements. A masking operation with a thresholded input image was then done on the extracted round components. The small regions and the areas without any holes were eliminated from the final resulting manhole image.

Detection of obscure and textured circular objects were challenges faced by conventional methods of object detection. In [30] (2009), this drawback was overcome without the cost of learning patterns. This method was valid for even images with inhomogeneous contrast and noise as it analyzed the separability and uniformity of intensity distributions using the Bhattacharyya coefficient filter, rather than the conventional method at the time, i.e., analyzing the difference in intensity levels of the object interior and surroundings. Separability in this paper is defined such that it can handle image feature distributions that are not normal distributions.

In [17] (2014), manhole cover detection using vehicle-based multi-sensor data combining multi-view matching and feature extraction was developed. Close range images using GPS/IMU and LIDAR data were obtained. It involved two main steps – edge detection and texture recognition. Scene segmentation to eliminate cars and pedestrians was done, and on the segmented data, a custom edge detection algorithm based on Canny edge detection [6] (1986), which was sensitive to arcs and ellipses, was used. Arc-containing regions were fitted to an ellipse.

An algorithm for automatic recognition of manhole covers based on images from the Mobile Mapping System (MMS) was proposed in [8] (2016). The images were collected using the MMS and preprocessed by image enhancement using gray-scale transformation and filtering technology, followed by the double threshold method in Canny operator edge detection. Hough transform based on the rough localization of ellipse geometry was used to locate the accurate positions of manhole covers in the preprocessed images.

In [50] (2020) the author concentrated on the detection of manhole covers using texture-based image segmentation and elliptical fitting. To extract textural features, the Laplacian of Gaussian filter's performance was contrasted with that of the Gabor filter. To divide the pixels in the image into distinct regions, the K-means technique with sum of square error was employed, and the least-squares approach was utilized to process the ellipse fitting.

In all of the above papers, only the localization of a manhole in the image is provided. Additionally, classification of manholes or differentiating between the various objects within the image is not attempted. The true sense of object detection as we know it now, is classification+localization. The above papers do not focus on this and thus, our paper attempts to fill this research gap by implementing a classical computer vision approach to manhole *detection* in road surface images. Our approach models conventional deep learning based approaches by providing a bounding box and coordinates for localization, and classifies the manholes too. Multiple bounding boxes are predicted in the localization step as in deep learning models, but the mean of the best fits is taken for the final classification. Additionally, like in all of the papers reviewed above, we extract the shape of the manhole from the image using ellipse fitting.

2.2. Machine and deep learning approaches for manhole detection in road-surface images

In [43] (2011), a multi-view method implementing 2-D and 3-D techniques for manhole mapping based on vision and GPS was presented. The position and inclination of the ground plane were estimated to generate front-to-parallel 2-D views. Single-view processing involved the application of a cascaded framework composed of mean-shift color segmented area, aspect ratio, intensity variance, radial symmetry, and texture-based filters to the 2-D views. The object detection system used for identifying manholes in single-view processing was Local Binary Pattern feature vectors [31] and Discriminatively

Trained Part-Based Model [14]. Multi-view processing involved fusing and grouping the results of single-view processing into 3-D hypotheses which were then fed into a graph-cut segmentation filter, and finally used for accurate localization of the manholes.

Automated detection of manholes using Mobile Laser Scanning (MLS) data was put forward in [52] (2015). The road surface images were segmented by detecting curbs in the images and using them as reference points for segmentation. These images were converted to raster images with georeferencing and intensity information using Inverse Distance Weighted (IDW) Interpolation. The high-order features of images were depicted by a multilayered feature generation model which was built on a vision based deep learning model. A random forest model was then trained to learn how these features are mapped to the probability of existence of manhole covers at specific locations. The manhole covers were then detected in the previously rasterized images using both models.

In [51] (2019), deep learning was suggested for the autonomous extraction of tiny objects in urban environments. A Mobile Mapped System was used to gather a dataset for Urban Element Detection (UED) that included manholes, milestones, and license plates. The faster R-CNN framework was tuned for small object identification, and a feature extraction CNN network named SlimNet with six convolutional layers and three max-pooling layers was developed. The performance of these networks on the collected dataset was compared to other existing deep networks. The findings of this paper concluded that the SlimNet model had the highest accuracy.

In [4] (2019), the Automated Localization of urban drainage infrastructure from public-access street-level images was done. A dataset of manhole and storm drain images was captured using the Google Street View API and annotated. The Faster R-CNN deep learning meta-architecture with Resnet 101 as the feature extractor backbone was tested on this dataset. Localization was done to project the coordinates from image space to geographical coordinates.

Mapping manholes using the deep learning method RetinaNet in road-level RGB images was done in [39] (2020). ResNet-50 and ResNet-101, being the two different feature extractor networks for the RetinaNet method, were used to experimentally test the method. The results of this test were then compared with the Faster R-CNN method. However, the findings of the paper concluded that the RetinaNet method was far more effective than the Faster R-CNN method for mapping manholes.

A method was proposed in [10] (2020) to convert RGB image data and extracted contours using an object detection algorithm. Pavement distress was classified using the YOLOv3 (You Only Look Once) algorithm which is a one-stage detection algorithm that does not need the region proposal phase. A large-scale dataset was prepared, containing images taken in various weather and illumination conditions. The image data was analyzed effectively by using Average Precision (AP) as the indicator on the data.

In [15] (2021), two deep learning techniques were implemented for automated pavement distress detection and classification, namely Faster Region-based Convolutional Neural Networks (R-CNN) and YOLOv3. These deep learning frameworks were trained on a dataset the authors collected and validation accuracy was indicated using Average Precision (AP) and Receiver Operating Characteristic (ROC) curves. By contrasting the suggested model with manual quality assurance and quality control (QA/QC) results received on automated pavement data, the models were assessed.

In all of the above papers, only convolutional neural network based architectures have been researched for the task of manhole detection. In our paper, we also focus on manhole detection with vision transformers as they have gained popularity for object detection tasks. Additionally, the networks modeled in the above papers have not been tested for the detection speed. Since in this paper we attempted the task of manhole detection for autonomous vehicles, the computational power requirement and speed is a significant factor we took into account. This is the main reason why we attempted YOLOv3 and YOLOv3-Tiny object detection networks in our paper.

2.3. Theoretical review of object detection algorithms

Since we adopt deep learning based object detection approaches, we also include a review of the state of the art object detection models. The three main approaches towards object detection are:

1. Classical Computer Vision,
2. Convolutional Neural Networks,
3. Vision Transformers.

We attempt networks from all three of these categories in our paper.

Localization algorithms using classical computer vision have been explored to a large extent as seen from all the previous papers reviewed. The following papers give examples of classical computer vision being used for object localization and classification including bounding box calculation. In [26], the authors use a variant of Hough Transform for localization and Machine Learning for classification to perform object detection. Max Margin Hough Transform was used for localization and bounding box prediction, and classification was done using an SVM based classifier. In [20], classical computer vision was combined with machine learning to create a fully automated hybrid cell-detection model named CIRCLE. The images were first processed to extract various tiles from them. MaskRCNN was then applied to them to detect the cells. These works encouraged us to test classical computer vision approaches especially to focus on the detection speed for autonomous vehicles. However, we do not follow the same methodology for object detection as the above two papers; instead we propose a novel pipeline.

Convolutional Neural Networks have been popular in object detection for a long time now. The YOLO algorithm [35] is popular and YOLOv3 is the model we chose

for our research. YOLOv1 [35] is a one-stage detector which uses the Darknet framework [2] and is trained on the ImageNet-1000 dataset [13]. It splits a given image to a grid of $S \times S$ cells. For every cell in the grid, it computes confidence for n bounding boxes. The predicted result is encoded into a tensor of dimensions $S \times S \times (5n + p)$, wherein the input image is divided into $S \times S$ sub-images, the $5n$ term corresponds to five attributes of the bounding box that must be detected (center coordinates, height, weight, and confidence score). The p term represents the probability of the object in the image belonging to a particular class. YOLOv1 had difficulty with detecting small objects and when the dimensions of the testing images varied when compared to the training images [1]. YOLOv2 [36] improves upon this by introducing batch normalization in every convolutional layer. YOLOv3 [37] further improves upon this by using independent logistic classifiers for multilabel classification instead of multiclass classification when using softmax. This improves the model as using softmax imposes the assumption that each box has exactly one class which is often not the case. The key novelty of the YOLOv3 algorithm is that it makes its detections at three different scales. YOLOv4 [3] furthers upon YOLOv3 to introduce a new architecture with a backbone, neck, dense prediction, and sparse prediction. The backbone and dense prediction networks are similar to that of YOLOv3 (the backbone is changed to Cross Stage Partial Network (CSPNet) [47, 48]), and the neck is a novel idea to add layers in between the backbone and dense prediction block. The layers added to the neck are a modified Path Aggregation Network (PANet) [25], a modified spatial attention module, and a modified spatial pyramid pooling, which are all utilized to combine the data in order to increase accuracy. The CSPDarknet53 backbone [47] eliminates repetitive gradient information in big backbones and incorporates gradient change into a feature map that speeds up inference, improves accuracy, and shrinks the size of the model by reducing the number of parameters. YOLOv5 [18,19] utilizes the same CSPDarknet53 as backbone. The Path Aggregation Network in YOLOv5 is different and adopts a new feature pyramid network (FPN) that includes several bottom up and top down layers. The model's low level feature propagation and localization precision are both enhanced by this PANet. The localization accuracy of the object is increased because of PANet's improved localization in lower levels. In YOLOv7 [46], the PAnet is replaced by Extended Efficient Layer Aggregation Network (EELAN) which uses group convolution to enhance the features learned by different feature maps and improve the use of parameters and calculations. In addition, compound model scaling is used. This is scaling the width (number of channels) and depth (number of layers) in coherence for concatenation based models. A summary of the architecture of these models can be found in Table 1.

Transformers were first proposed for Natural Language Processing tasks in [45]. In [9], the authors propose a Vision Transformer called ViT which closely models the original transformer architecture of [45] as closely as possible for image classification tasks. Images are first flattened into 2D patches to be passed to the transformer's encoder network.

Tab. 1. Table from [29] with an additional column added for YOLOv7.

	YOLOv3	**YOLOv4**	**YOLOv5**	**YOLOv7**
Neural Network	FCNN	FCNN	FCNN	FCNN
Backbone Feature Extractor	Darknet-53	CSPDarknet53	CSPDarknet53	CSPDarknet53
Loss Function	Binary Cross Entropy Loss	Binary Cross Entropy Loss	Binary Cross Entropy Loss and Logits Loss Function	Binary Cross Entropy Loss
Neck	FPN	SSP and PAnet	PAnet	EELAN
Head	YOLO Layer	YOLO Layer	YOLO Layer	YOLO Layer

The transformer's encoder consists of alternating layers of Multi-headed Self Attention (MSA) and Multi Layer Perceptron (MLP) blocks. Layernorm (LN) and residual connections are applied respectively before and after every block. The MLP contains two layers with a GELU non-linearity. Due to its accuracy on ImageNet, in [7], a Vision Transformer was used for object detection to develop the DETR model. The DETR model uses the feature maps extracted by a CNN backbone as input for the transformer encoder-decoder architecture for transforming feature maps to features, followed by a single feed forward neural network for prediction. It uses a bipartite matching loss function for matching between the predicted tokens and ground-truth objects. In [24], instead of using a CNN backbone for object detection as done in [7], the authors explore using a plain Vision Transformer like vanilla ViT as a backbone for object detection. In [12], the authors propose YOLOS, an object detection model which uses plain Vision Transformers ViT and DeiT [44] as the backbone. While YOLOS chose a Transformer with an encoder-only architecture similar to ViT, DETR used a Transformer encoder-decoder architecture. For each encoder layer, YOLOS always examines a single sequence without making a distinction between the tokens in terms of operations, where the tokens are the learnable embeddings in the image.

For our work, we chose YOLOv3 and YOLOv3-Tiny as the networks to compare to YOLOS. We wanted to compare the performance of a *tiny* object detection model as it can be used for autonomous vehicle research. YOLOv4 and YOLOv3 have *tiny* object detection networks and as seen in the above figure, the authors of YOLOS also released a YOLOS-Tiny model. That is why these networks were chosen for our research work. Also, YOLOS is a completely transformer based architecture with a transformer backbone, and YOLOv3 is a completely CNN based architecture which makes it a good point of comparison.

3. Methods

3.1. Functional Block Diagram

Fig. 1 and Fig. 2 describe the proposed methodology's general flow. A visual explanation of some of the methodology's key decision points is provided by the flowchart. This modular structure makes it simple to plan for future changes to functionality and flow that will boost efficiency and allow integration with additional methods.

3.2. Dataset

We mainly used road surface images for this study due to its application in autonomous vehicles. However, few close-up aerial images were added to the dataset as well. Also in support of road-surface images, in [4] it was indicated that street-level imagery could provide useful information to identify manholes that could not be detected in aerial images. In this dataset, we focus only on round manhole covers. This is as most manhole covers are round, for the purpose that orientation when placing the cover is not an issue. At the same time since manholes weigh around 250 pounds [11], it is easy to roll them in case of replacement.

We have used the dataset publicly provided in [38] and described in [39].

A total of 1032 images were collected and annotated. Data augmentation in the form of 90° rotation, and saturation value change were performed in the dataset to obtain a dataset containing 2673 images. This was carried out to balance the number of images per class, avoid overfitting, and enhance the deep neural networks' performance during model training. The images were split into three different classes in an unbalanced fashion: closed manhole improperly closed manhole open manhole

Since we used the same dataset for classification and object detection, it is available in two different formats. In the format for classification, all the images have been divided into folders based on the class label without any annotation file as required in the classical computer vision object detection method. The drawback of this dataset is that images which have multiple objects each of different classes, the objects cannot be classified separately in the same image. All of the images were manually annotated in the format for deep learning-based object detection by marking rectangles (bounding boxes) around the manholes and categorising each rectangle by the corresponding class. This was done using the `labelImg` tool [23]. Images which have multiple objects, each of different classes, can be classified separately in the same image.

For training, these images were divided into three groups for training, testing, and validation. The train-test-validation split is done randomly with no manual intervention. As the dataset is minimal already, we increased the number of training samples to contribute to a more robust evaluation. The number of images in each of the sets are as follows:

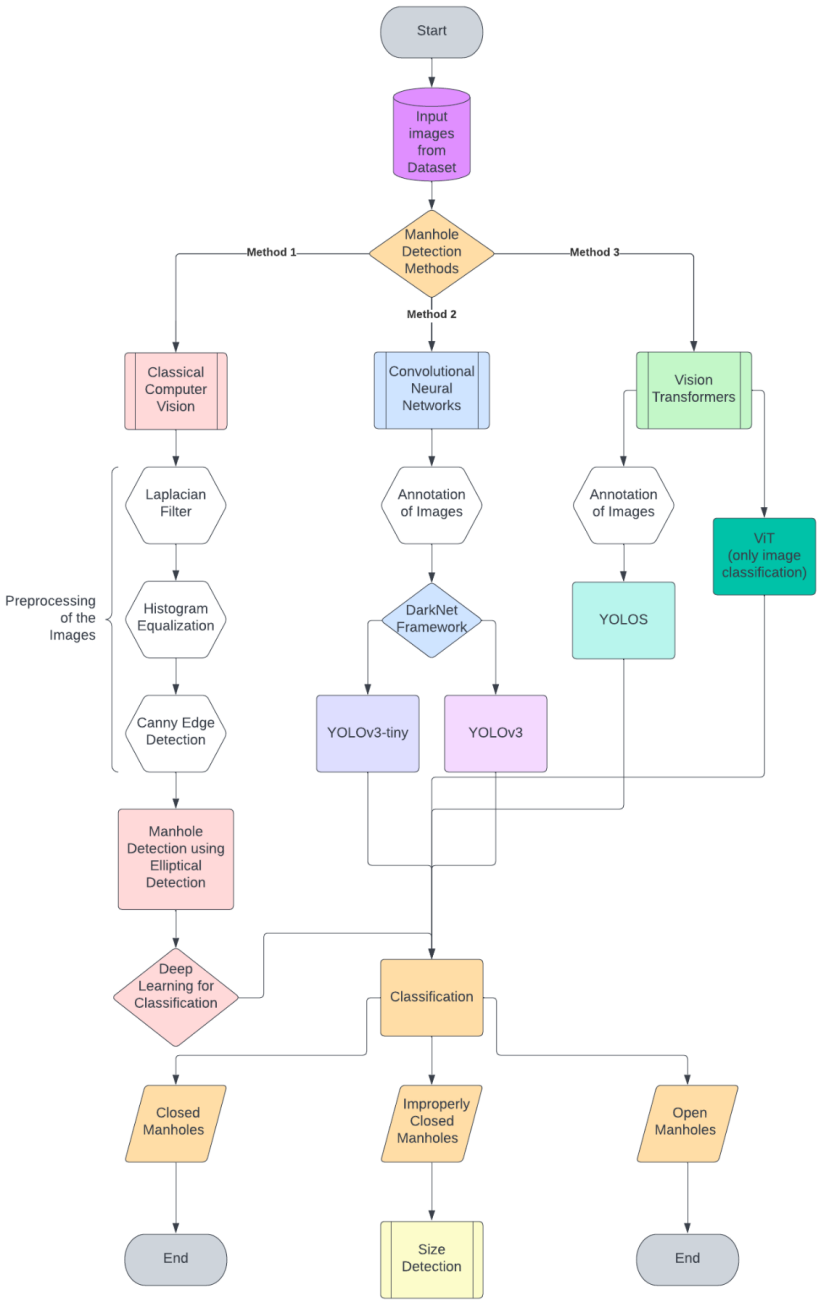


Fig. 1. Block diagram of the methodology

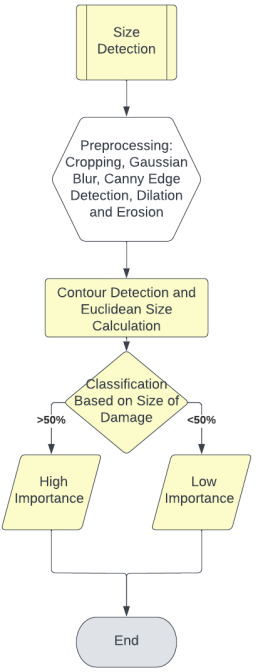


Fig. 2. Block diagram of the size detection algorithm

1. Training – 2469
2. Testing – 71
3. Validation – 133

Fig. 3 depicts an example of an image which has been annotated with a bounding box drawn around the manhole.

3.3. Methodology

3.3.1. Object Detection

In Section 2 in the Review of Literature, we elaborated on the architectural differences between the three main approaches towards object detection. As described in the review of literature, we implemented classical computer vision approaches, CNN-based YOLOv3 and YOLOv3-Tiny, and Vision Transformer based YOLOS. We evaluated the performance of each of these models on our dataset to determine the best for manhole detection and classification specifically for autonomous vehicles. A summary and comparison of the different methods used in this paper is available in Table 2.

Fig. 3. Example of an annotated image.

Classical Computer Vision approach to manhole detection

In this method, we propose a novel pipeline for manhole detection in road-surface images. This method was attempted as it does not require specialized hardware like a GPU and the computational complexity of the algorithms is not high especially in comparison to deep learning approaches. Since we are evaluating the performance of the algorithms for self-driving vehicles, we expected this method to be advantageous. In addition, the dataset collected is minimal for deep learning tasks so we proceeded with this approach. The code for this approach was implemented with MATLAB.

1. **Pre-processing**

   In classical computer vision approaches, the filtration and preprocessing stage is an essential part of object detection. We combined multiple approaches in order to get a high level of accuracy. For filtering, we employed the Laplacian filter [16]. This filter was chosen since it preserved the edges while reducing the noise. This filtering was followed by histogram equalization to improve the overall contrast of the image as low contrast regions are brought closer to the average contrast value. Next, Canny Edge Detection [6] was done to detect the edges of objects in the images. We applied this technique to extract the morphological information from the images while also reducing the data to be processed before performing manhole localization.

   Fig. 4 shows an example of how an image looks after preprocessing.

2. **Manhole Localization**

   The geometrical circular detection filter is based on the approach proposed in [30] and adapted in [32]. However, since the manholes in road surface images have a mostly

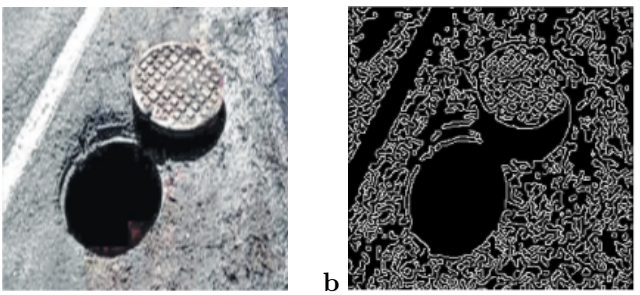


Fig. 4. An example of the effect of preprocessing. (**a**) Original image; (**b**) pre-processed image.

elliptical shape, we used an elliptical filter. The code for the ellipse detection is adapted from [41]. It can be used for circle detection also. The theory behind this approach is that for an ellipse, there are five unknown parameters, (x_0, y_0) for the center, (a, b) for the major and minor axes, and α for the orientation. In [49], the author proposes a method using only a 1D accumulator array to accumulate the length of the minor axis of the ellipse. Using this method, the four coordinates of the major and minor axes of the ellipse were calculated.

The shapes that were the best fit as per the definition of an ellipse were taken. In the case of more than one best fit, the mean of the best fits was taken to be the final ellipse. Once the ellipse was identified, two rectangles were identified taking each of the major and minor axes of the ellipse as diagonals. In order to make sure that no part of the ellipse is cut out, the final rectangle was taken so that it bounds both of the rectangles created. The coordinates of the rectangular bounding box drawn around the manhole in the images are obtained. Multiple manholes in the same image can be localized this way.

A mask was created such that the pixels in the area covered by the final bounding rectangle all had a value of 1, and those not covered by the rectangle all had a value of 0. The original image and the mask were combined, finally showing only the manhole. There were some cases in which the model failed to identify a rectangle showing only the desired part of the image. To ensure that this was not a problem, we allowed the original preprocessed image itself to be taken as the final product in such a case wherein the image was already cropped enough and the manhole was in the region of interest. Fig. 5 shows examples of ellipses drawn around manholes in each case: closed manhole, open manhole and broken manhole.

In our attempt to ensure that as much as possible of the manhole is retained while the unnecessary information is deleted, we had chosen to take the mean of the best fits (as many possible ellipses were found) as the final ellipse, and used these coordinates to draw a bounding box. Using the mean of the best fits may have resulted in the

Fig. 5. Samples of the ellipse drawn by the algorithm around each class: (**a**) closed, (**b**) broken, and (**c**) open manhole.

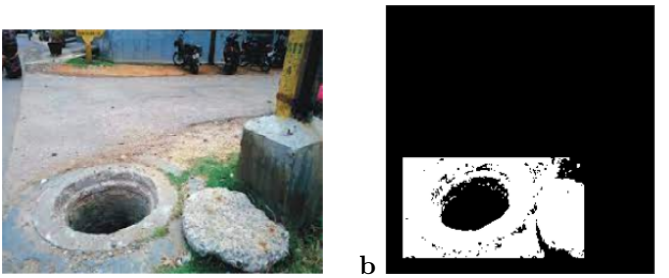


Fig. 6. Bounding box drawn around the manhole using the ellipse coordinates. (**a**) Original image; (**b**) manhole detected (the bounding box is the region with white background).

inclusion of some extra pixels, as seen in Fig. 5; the ellipses drawn were not perfectly accurate to the edge. However, this does ensure that none of the necessary information for training the deep learning model was deleted, while a majority of the unwanted material was effectively removed.

Instead of drawing an ellipse around the manhole as done in Fig. 5, we can use the coordinates of the ellipse directly to obtain the coordinates for the bounding box. In the algorithm we used, the pixels inside the bounding box are left unchanged while the values of the pixels outside the bounding box are all equated to zero.

The resulting image after the bounding box is defined is shown in Fig. 6 for another manhole from our dataset. The left and right most x-coordinates and top and bottom most y-coordinates were calculated from the major and minor axes coordinates of the ellipse defined. As seen in the image, the pixels inside the box are left as they were while all the other pixels in the image were made black. The time taken for localization of all of the images was around 6 hours on a CPU.

3. **Manhole Classification**

The images obtained after localization were taken for the deep learning model. Since it was for simple classification, we used a basic Convolutional Neural Network architecture. We used 3 convolutional layers, 2 max pooling layers followed by a fully

connected layer. ReLu was taken as the activation function for the convolutional layer and softmax for the final classification. Batch Normalization is used after every Convolutional Layer as used in YOLOv3 [37]. The number of epochs was set to 100 and the number of iterations per epoch was set to 6. The training time was 68 minutes.

Convolutional Neural Networks approach to manhole detection

In this method, we used YOLOv3 and YOLOv3-Tiny for manhole detection. As the performance of the classical computer vision algorithm could be improved greatly, we implemented a convolutional neural network based approach notwithstanding the computational intensity. There are many other state-of-the-art object detection models, however we chose to analyze the performance of YOLOv3 as it has a YOLOV3-Tiny model in conjunction. We hypothesized that the YOLOv3-Tiny model would especially prove useful for self-driving technology as it has a higher FPS (frames per second) rate as per [35]. In addition, YOLOv3-Tiny was attempted as it is not computation intensive and is a small model meant for constrained environments. The code for this method was implemented in Python.

YOLO is a deep Convolutional Neural Network based one-stage object detector. YOLO only looks at an image once to predict whether the object is present and where it is located in the image. YOLO implicitly encodes contextual and visual information about classes as it views the entire image during training and test. Object detection in YOLOv3 is framed as a logistic regression problem to separate the bounding boxes in the image and associate class probabilities with each bounding box [35]. A single neural network predicts the coordinates of the bounding box using dimension clusters as anchor boxes. YOLOv3 uses K-means clustering on the dataset to determine bounding box priors automatically rather than manually. The probability of the object class and the confidence of the object in the bounding box is captured in one evaluation itself. The probabilities for each class are calculated using independent logistic classifiers rather than the softmax function; thus, each bounding box can belong to several classes. Thus, as the detection pipeline is a single network and end-to-end optimizations is performed to improve detection, the unified architecture of YOLOv3 is extremely fast and robust running at 100 FPS (frames per second). YOLO v3 uses upsample, downsample, and fusion methods to independently detect objects on multiple scales of fusion feature maps. This method is especially effective in detecting small and near-distance target objects; thus, it's suitable for manhole detection.

We adopted the DarkNet-53 (53 convolutional layers) neural network framework as the backbone for YOLOv3. We adopted transfer learning, so our models' weights were initialized with weights from the network trained on the MS COCO dataset [37]. We adopted this approach in our project as transfer learning enables training of neural networks with lesser data, the accuracy is high, and the training time is reduced. We used the source code available in [2] for our implementation. The model was trained and tested on Google Colaboratory with an NVIDIA Tesla T4 Graphics Card (2560 Compute

Unified Device Architecture (CUDA) cores and 16 GB graphics memory). The number of iterations was set to 6,000 (as specified in [2]) for both YOLOv3 and YOLOv3-Tiny. The classes, number of filters, steps, jitter, etc., were also customized to our requirements in the config file. As it's an open-source framework, we tuned a few hyperparameters to customize the network to our task as documented in [2]. However, we adopted *early-stopping* to prevent overfitting. Also, due to the time it takes to run the program, and because of the desirable mAP value obtained, we stopped training the model at 3000 iterations for both networks. The only difference in training between the two models YOLOv3 and YOLOv3-Tiny was the configuration file used. The YOLOv3 model ran for 3035 iterations. YOLOv3 took 5 hours and 15 mins for it to train, even with a GPU that supports fast computation. Thus, it's a very time-consuming process to train. YOLOv3-Tiny ran for 3020 iterations.

Vision Transformer based approach to manhole detection

In this method, we used YOLOS [12], a Vision Transformer based model for object detection. Vision Transformers have recently gained a lot of popularity in object detection tasks, according to [7]. Out of all of the Transformer based object detection models, we chose YOLOS as it uses a Transformer as the backbone. Also since it is a single sequence based architecture, we wanted to compare it to the YOLO which is also a one-shot object detector. The code for this method was implemented in Python.

YOLOS uses a *plain vanilla* Vision Transformer (ViT) as its backbone [12]. The difference between ViT and the backbone for YOLOS is that YOLOS drops the CLS token used for image classification and adds 100 randomly initialized detection tokens [DET] to the input patch embedding sequence. The CLS token indicates that the training task is classification. The [DET] token is a learnable embedding for object binding. Position embeddings are added to all of these input tokens to retain positional information. Another difference between ViT and YOLOS' backbone is that the image classification loss used in ViT is replaced with a bipartite matching loss to perform object detection similar to DETR [7]. During training, YOLOS produces an optimal bipartite matching between predictions from the one hundred [DET] tokens and the ground truth objects. During inference, YOLOS directly outputs the final set of predictions in parallel.

The randomly initialized detection [DET] tokens are used as substitutes for object representation. This is done to avoid inductive bias and any prior knowledge of the task that can be introduced during label assignment. When YOLOS models are fine-tuned on the COCO dataset, an optimal bipartite matching between predictions generated by [DET] tokens and the ground truth is established for each forward pass. This serves the same purpose as label assignment but is completely unaware of the input 2D structure, or even that it is 2D in nature. These steps to reduce the inductive bias are imperative as some of the YOLO algorithms' (CNN based YOLO family) performance reduces when the dimensions of the testing images vary when compared to the training images, as stated in [1].

We trained YOLOS for 150 epochs for batches of 8 and monitored the validation loss when training. The images were tested for a box confidence score of 0.2. The model was trained and tested on Google Colaboratory with an NVIDIA Tesla T4 Graphics Card and it took around 9 hours to train. The accuracy of YOLOS is not very accurate, as stated in [12], and we noticed this too. Thus, we attempted plain Vision Transformer ViT with no hierarchical backbone as the authors of [24] pointed out that vanilla ViT could be used as a backbone for classification in object detection networks with good accuracy. We tried ViT on our dataset to check if Visual Transformers were promising for manhole classification as YOLOS uses this as its transformer backbone. We trained the ViT model for 30 epochs in a batch size of 20. The total time taken for training was around 15 minutes with a Tesla T4 GPU.

3.3.2. Size Detection

The manholes which were detected as improperly closed through the object detection model were run through a size detection algorithm. The algorithm detects the size of the hole in a broken manhole with respect to the size of the manhole itself in an image. We used classical computer vision using python and OpenCV for size detection.

1. **Preprocessing**

   The manually annotated images from the dataset were first converted from the YOLO to the COCO annotation format. Using those coordinates, we automated the process of cropping the manholes identified as improperly closed around their bounding box to avoid unnecessary visual information.

   The cropped images were then Gaussian blurred. This step was done to reduce the extraneous visual information in the image. The image was then converted to grayscale to apply Canny edge detection which enabled us to outline the manhole and the broken hole. After this, the image was *opened* (dilation+erosion).

2. **Algorithm**

   Canny edge detector detects all of the edges in an image, but we need only the manhole and the broken hole. Hence, we applied contour detection methods. This method was preferred over ellipse detection methods like the Hough Transform as a broken manhole is not elliptical in shape anymore. Additionally, the broken part of the manhole is also not necessarily elliptical in shape. The manhole was identified by detecting the maximum contour in the image. We draw a bounding box around the contour of the manhole. Using the coordinates of the ordered bounding box, we compute the midpoint of the top left and top right coordinates and the midpoint of the bottom left and bottom right coordinates. We calculate the Euclidean distance between the midpoints using the distance formula in terms of pixels. We follow the same steps for all of the contours found in the image except the max contour. If any of the contours found in the image are greater than 50% of the max contour (i.e. the entire manhole), we classify that manhole as a *High Importance* manhole in need of

Tab. 2. Summary and comparison of the object detection methods used in this paper.

Approach	Automated Method	Localization Method	Classification Method	Pros of the Approach	Cons of the Approach
Classical CV	Automatic Annotation (just classified images into classes)	Classical CV	Custom Neural Network	– Localization is done in the shape of the manhole before drawing the conventional bounding box. – The images are pre-classified into the classes. Automated annotation of the dataset. – Time taken was around 7 hours for all of the images.	– Not very accurate. – Localization works for multiple manholes in the same image. Cannot classify each of the manholes in the localized image with the deep Neural Network however.
YOLOv3 and YOLOv3-Tiny (CNN)	Manual Annotation	Simultaneous localization and classification (CNN)	Simultaneous localization and classification (CNN)	– Accurate. – Can localize and classify multiple instances of manholes in the same image. – Gives the probability of the object class in the box. – Time taken to train was 3, 6, 9 hours respectively to train the 3 models.	– Requires specialized hardware (GPU). – Annotation is manual.
YOLOS	Manual Annotation	Simultaneous localization and classification (Vision Transformer)	Simultaneous localization and classification (Vision Transformer)		

immediate attention by the concerned authorities. The rest of the improperly closed manholes are classified as low importance. Thus the method of ranking according to safety is:

Open manholes > Improperly closed manholes whose broken parts are greater than 50% of the entire manholes > Rest of the Improperly closed manholes.

Size detection did not take a lot of time. Contour detection methods are fast and have already proved useful in developing computer vision algorithms for real-time autonomous vehicles [34].

4. Results

4.1. Object Detection

Classical Computer Vision approach to manhole detection

The metrics used to evaluate the classification accuracy of the neural network were Testing/Validation Accuracy and Loss. An image of the graph at the end of 100 epochs and 6 interactions per epoch is shown in Fig. 7. According to the graph, Testing/Validation Accuracy is 63.23%. The images chosen for evaluating the accuracy were randomly selected 150 images from the validation and test datasets. As there was no hyperparameter tuning and the the images from the validation dataset were not shown to the model more than one time, this served as the Testing Accuracy though the images from the validation dataset were used. The model was trained as the accuracy percentage increased and the loss percentage decreased. Training accuracy finally reached 100%.

Convolutional Neural Networks and Vision Transformer approach to Manhole Detection

The performance of DarkNet YOLOv3, YOLOv3-Tiny, YOLOS (for object detection), and ViT (for just classification) were assessed by precision–recall curves and the average precision (AP). The Intersection over Union (IoU) was calculated in order to evaluate the precision and recall. According to well-known competitions in object detection, a true positive (TP) is for IoU ≥ 0.5 and a false positive (FP) for IoU < 0.5. Based on the above metrics, precision (P) and recall (R) are estimated. The average precision (AP) is estimated by the area under the precision–recall curve. The Mean Average Precision (mAP) was used as the primary indicator to analyze the accuracy of the models as seen in Fig. 8. The AP is calculated for each of the classes and it is averaged over all of the categories to get the mAP. We trained the models while the mAP percentage increased and the validation loss decreased. The mAP percentage value was calculated during training. The loss curves also indicate accuracy, however, mAP is considered to be a better indicator [37].

The weights files were generated for every 1000 iterations. The best weights obtained were saved and can be used for further testing and manhole classification of all 3 object detection models. At some points in the graph even though the mAP percentage reduces, we still continued training as the recall values were still increasing [35]. The average precision (AP [%]) and its mean values (mAP [%]) obtained from the area under the curve for both of the models are illustrated in the above figure. Table 4 displays the results at an IoU cutoff at 0.5 (AP50). The FPS rate was calculated as given in [40] with a batch size of 1. The size of the images in the image stream was taken as 256×256 for testing purposes.

Thus we realized that the YOLOv3 model had the best accuracy but YOLOv3-Tiny

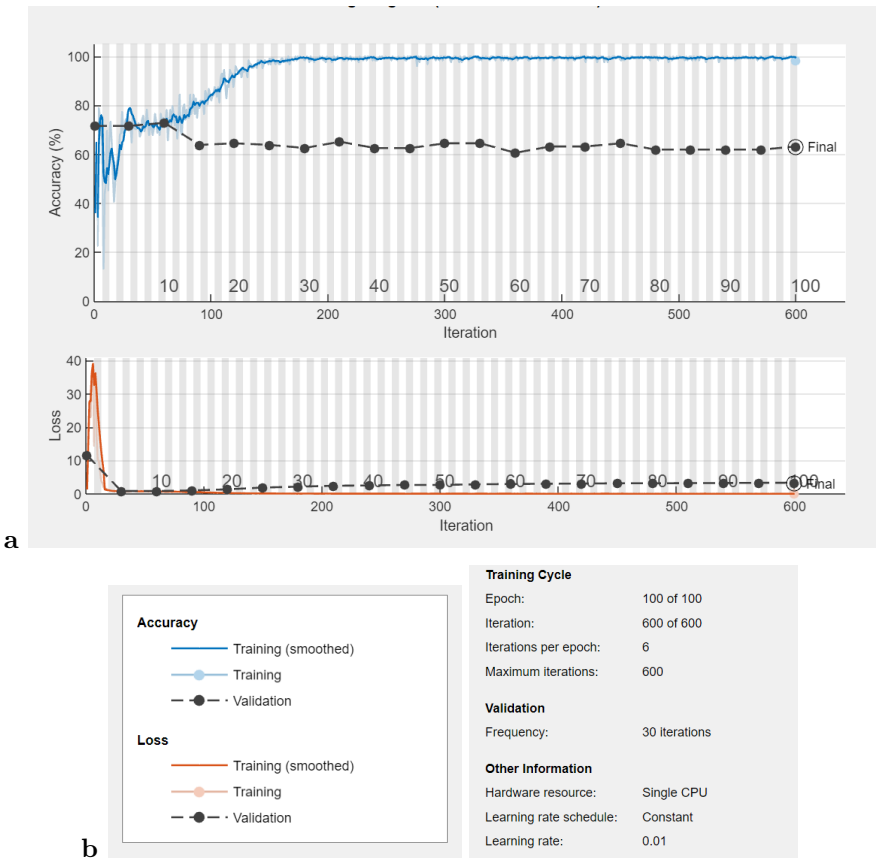


Fig. 7. (**a**) Training and validation accuracy and loss curves for the classification network of the classical CV approach. (**b**) Key for (a) and other information.

seems like the best model for manhole detection from the autonomous vehicles standpoint. Though the accuracy of the YOLOS model is not large, the ViT classification network gives good results. This indicates that further research into object detection models using transformers could prove promising. In terms of the computational requirement and speed too, CNNs and classical computer vision could be used.

Fig. 11 and Fig. 12 show examples of detected and classified manholes.

Since the accuracy of YOLOS was not high, we trained ViT on the same dataset just to evaluate the classification accuracy. The results in the form of the confusion matrix are shown in Fig. 13.

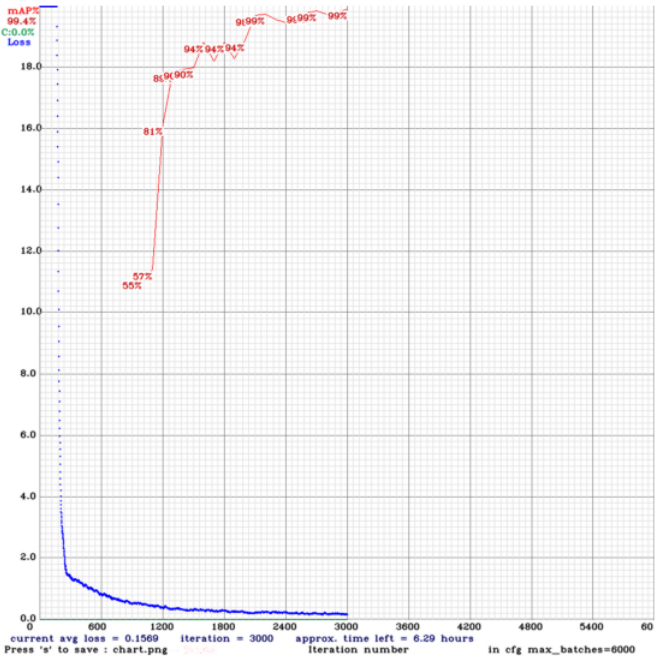


Fig. 8. mAP graph for YOLOv3.

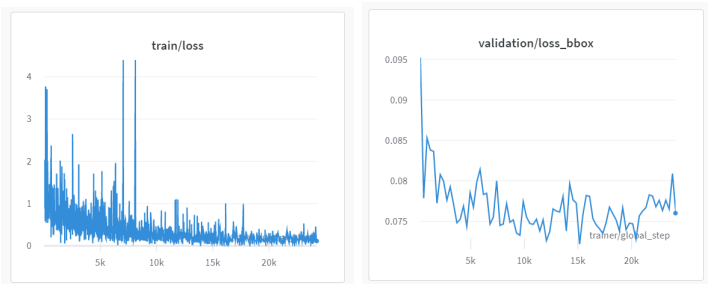


Fig. 9. Loss Graph for YOLOS.

4.2. Size Detection

We tested the size detection model on 26 improperly closed manholes which contained partially open manholes and broken manholes. We also included 2 open manholes and 2 closed manholes for testing purposes. The evaluation of this system model was done by manual classification as in, we cross verified if what we determined as high importance matched the size detection model's output. We determined that out of the 26 improperly

Tab. 3. Results of object detection models tested.

Network	Confidence Threshold	Precision	Recall	F1 score	mAP@0.5 or Accuracy	FPS rate with GPU	FPS rate with CPU
Classical CV	-	-	-	-	0.63	-	30
YOLOv3	0.25	0.98	0.98	0.98	0.994	33	-
YOLOv3 -Tiny	0.25	0.82	0.82	0.82	0.92	62	7
YOLOS	0.2	0.59	0.77	0.6	0.67	5	-
ViT*	0.2	0.84	0.81	0.83	0.91	-	-

* Not an object detection network, just tested for classification accuracy, thus does not have an FPS rate.

closed manhole images, 20 of them could be classified as high importance. Out of the 30 images the accuracy percentage of the method is as follows in Table 4.

Figs. 14, 15 and 16 depict the various predictions of the open manhole. If even one of the predictions was $> 50\%$ of the area, then it's a high importance manhole that needs to be fixed.

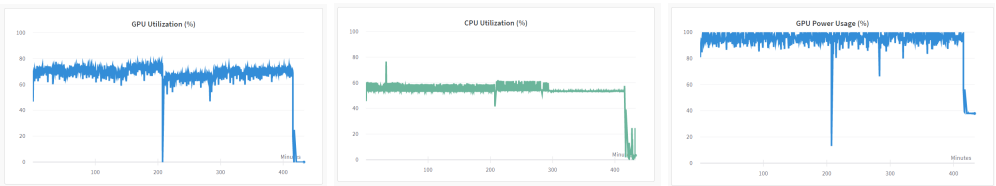


Fig. 10. GPU, CPU utilization and power consumption of YOLOS model.

Fig. 11. Test result of YOLOv3.

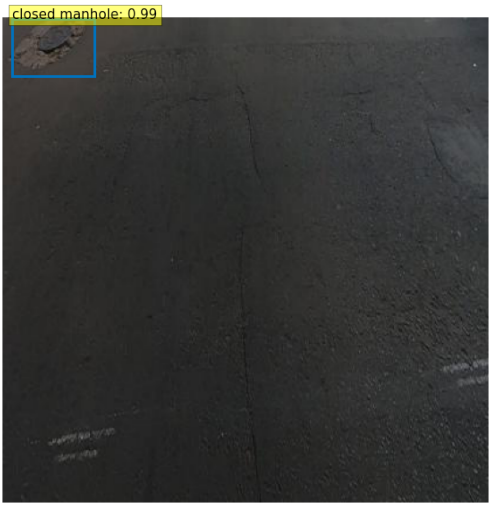


Fig. 12. Test result of YOLOS.

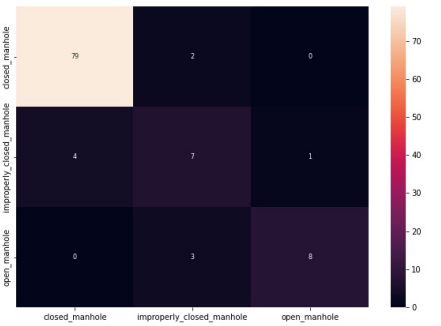


Fig. 13. Confusion matrix for ViT model.

Tab. 4. Results of size detection algorithm tested.

Class	**Size Detection Algorithm Accuracy**	**FPS Rate tested with CPU**
Open manholes	100%	30 fps
Closed manholes	100%	30 fps
Improperly closed manholes	90% *	30 fps

* There were 20 high importance manhole images from the 26 images of the improperly closed manhole class. 22/26 were classified as high importance.

5. Conclusion and Future Work

Including manhole detection models in obstacle avoidance and lane changing algorithms could improve the suitability of autonomous vehicles for use on the roads of developing countries. This research presented three different ways that autonomous vehicles may use for detecting manholes. We tested a classical computer vision approach involving image processing algorithms like Canny edge detection and ellipse detection. We also tested YOLOv3 and YOLOv3-Tiny to try Convolutional Neural Networks, and tested YOLOS and ViT to attempt vision transformer based approaches. The results of our research point to the conclusion that YOLOv3-Tiny would be the most suitable model for deploying on autonomous vehicles. Since self-driving vehicles have a drive-by-wire

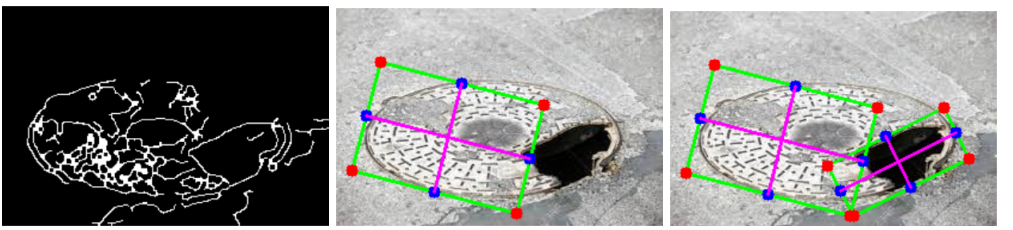


Fig. 14. Sample of a broken manhole.

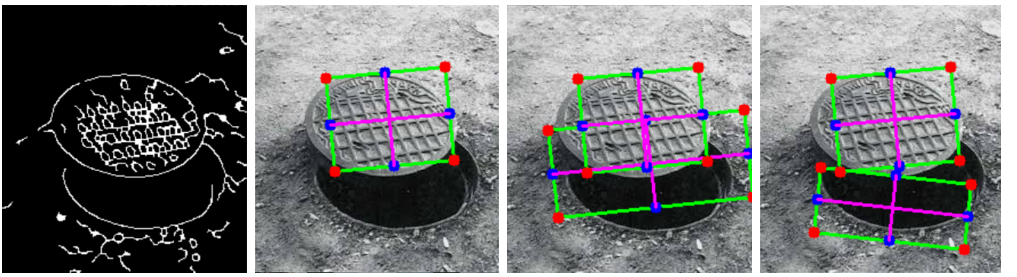


Fig. 15. Sample of a partially open manhole.

system and are equipped with cameras for computer vision, then the speed, computational complexity, and accuracy are the metrics that are taken into consideration when evaluating the three approaches to manhole detection. The GPU power consumption is low and FPS (frames per second) rate is very high for YOLOv3-Tiny, further reinforcing our conclusion. Vision Transformers also seem promising for this purpose with future research advancements.

In addition to manhole detection, our research presents a pipeline for size detection using filters and a contour detection method. The size of the hole in the broken/improperly closed manhole with respect to the manhole itself is determined and

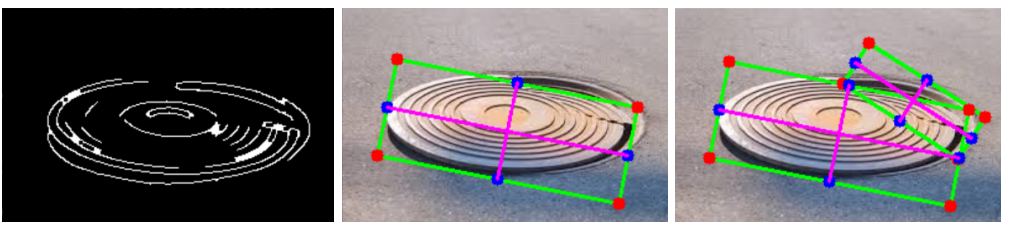


Fig. 16. Sample of a broken manhole that looks closed.

classified as high or low importance to determine a priority order for fixing them. Technology is designed to meet the requirements of different locations and the aim of science and development is to improve the standard of living in the places where they are used. This size detection model provides a method to not only detect manholes to avoid pavement distress and improve the standard of driving, but also enables the vehicle to aid in solving the problem of fixing an open, broken, or improperly closed manhole. With this in mind, we propose a future course of action. Autonomous vehicles are equipped with GPS and this system can be used to record the coordinates of the open or broken manhole along with its priority order from the size detection model. This information can then be used to alert the authorities for damage control.

In the future, we intend to work on aspects of research on autonomous vehicles that bring us one step closer to making self-driving vehicles a reality even in developing countries.

Acknowledgement

The authors would like to acknowledge BITS Pilani Hyderabad Campus for providing us an opportunity to carry out this project as part of Remote Sensing and Image Processing, under the guidance of Dr. K. Rajitha of the Civil Engineering Department.

Data accessibility statement

The data used in our work is accessible as open source at [33]. In our study many images from the dataset [38] described in [39] were used.

References

[1] P. Adarsh, P. Rathi, and M. Kumar. YOLO v3-Tiny: object detection and recognition using one stage improved model. In *6th Int. Conf. Advanced Computing and Communication Systems (ICACCS 2020)*, pages 687–694, Coimbatore, India, 6-7 Mar 2020. doi:10.1109/ICACCS48705.2020.9074315.

[2] A. Bochkovskiy. darknet. GitHub, 30 Oct 2021. `https://github.com/AlexeyAB/darknet`. [Accessed 17 Oct, 2022.

[3] A. Bochkovskiy, C. Y. Wang, and H. Y. M. Liao. YOLOv4: Optimal speed and accuracy of object detection, 2020. arXiv:2004.10934. doi:10.48550/arXiv.2004.10934.

[4] D. Boller, M. M. Vitry, J. D. Wegner, and J. P. Leitão. Automated localization of urban drainage infrastructure from public-access street-level images. *Urban Water Journal*, 16(7):480–493, 2019. doi:10.1080/1573062X.2019.1687743.

[5] Can self-driving cars run on Indian roads? *moneycontrol*. [Accessed 17 Oct, 2022]. `https://www.moneycontrol.com/news/driverless-cars/`.

[6] J. Canny. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(6):679–698, 1986. doi:10.1109/TPAMI.1986.4767851.

[7] N. Carion, F. Massa, G. Synnaeve, et al. End-to-end object detection with transformers. In *Proc. European Conf. Computer Vision (ECCV 2020)*, volume 12346 of *Lecture Notes in Computer Science*, pages 213–229, Glasgow, UK, 23-28 Aug 2020. doi:10.1007/978-3-030-58452-8_13.

[8] Z. Chong and L. Yang. An algorithm for automatic recognition of manhole covers based on MMS images. In *Proc. 11th Chinese Conf. Advances in Image and Graphics Technologies (IGTA 2016)*, volume 634 of *Communications in Computer and Information Science*. Springer, Beijing, China, 8-9 Jul 2016. doi:10.1007/978-981-10-2260-9_4.

[9] A. Dosovitskiy, L. Beyer, A. Kolesnikov, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv*, 2020. arXiv:2010.11929v2. doi:10.48550/arXiv.2010.11929.

[10] Y. Du, N. Pan, Z. Xu, et al. Pavement distress detection and classification based on YOLO network. *International Journal of Pavement Engineering*, 22(13):1659–1672, 2021. doi:10.1080/10298436.2020.1714047.

[11] L. Eliot. Manhole covers and self-driving cars. In: Self-Driving Cars. Podcasts by Dr. Lance Eliot, 9 Sep 2021. `https://ai-selfdriving-cars.libsyn.com/manhole-covers-and-self-driving-cars`. [Accessed 17 Oct, 2022].

[12] Y. Fang, B. Liao, X. Wang, et al. You Only Look at One Sequence: Rethinking transformer in vision through object detection. In *Advances in Neural Information Processing Systems (NeurIPS 2021)*, volume 34, pages 26183–26197. 2021. `https://proceedings.neurips.cc/paper/2021/hash/dc912a253d1e9ba40e2c597ed2376640-Abstract.html`.

[13] L. Fei-Fei, J. Deng, O. Russakovsky, A. Berg, and K. Li, editors. *IMAGENET*. 2021. [Accessed December 2022]. `https://image-net.org`.

[14] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part-based models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(9):1627–1645, 2010-09. doi:10.1109/TPAMI.2009.167.

[15] R. Ghosh and O. Smadi. Automated detection and classification of pavement distresses using 3D pavement surface images and deep learning. *Transportation Research Record*, 2675(9):1359–1374, 2021. doi:10.1177/03611981211007481.

[16] E. Hildreth and D. Marr. Theory of edge detection. *Proceedings of the Royal Society of London. Series B, Biological sciences*, 207(1167):187–218, 1980. doi:10.1098/rspb.1980.0020.

[17] S. Ji, Y. Shi, and Z. Shi. Manhole cover detection using vehicle-based multi-sensor data. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, XXXIX-B3:281–284, 2012. doi:10.5194/isprsarchives-XXXIX-B3-281-2012.

[18] G. Jocher, A. Chaurasia, A. Stoken, et al. ultralytics/yolov5: v7.0 - YOLOv5 SOTA Realtime Instance Segmentation. Zenodo, Nov 2022. doi:10.5281/zenodo.7347926.

[19] G. Jocher et al. YOLOv5 by Ultralytics. GitHub, 2020. `https://github.com/ultralytics/yolov5`. [Accessed 17 Oct, 2022].

[20] E. Karimi, M. Rezanejad, B. Fiset, et al. Machine learning meets classical computer vision for accurate cell identification., 28 Feb 2022. doi:10.1101/2022.02.27.482183.

[21] Y. M. Kim, Y. G. Kim, S. Y. Son, et al. Review of recent automated pothole-detection methods. *Applied Sciences*, 12(11):5320, 2022. doi:10.3390/app12115320.

[22] C. Kumar. Preventable deaths: In India, at least 2 die each day due to open pits & manholes. *The Times of India*, 25 Nov 2021. [Accessed 9 Oct, 2022]. `https://timesofindia.indiatimes.com/`

india/preventable-deaths-in-india-at-least-2-die-each-day-due-to-open-pits-manholes/articleshow/87917848.cms.

[23] Label Studio community (originally created by Tzutalin). LabelImg. GitHub, 23 Sep 2022. https://github.com/heartexlabs/labelImg. [Accessed 17 Oct, 2022].

[24] Y. Li, H. Mao, R. Girshick, and K. He. Exploring plain vision transformer backbones for object detection. In S. Avidan et al., editors, *Proc. European Conf. Computer Vision (ECCV 2022)*, volume 13669 of *Lecture Notes in Computer Science*, pages 280–296, Tel Aviv, Israel, 23-27 Oct 2022. doi:10.1007/978-3-031-20077-9_17.

[25] S. Liu, L. Qi, H. Qin, et al. Path Aggregation Network for instance segmentation. *arXiv*, 2018. arXiv:1803.01534v4. doi:10.48550/arXiv.1803.01534.

[26] S. Maji and J. Malik. Object detection using a max-margin Hough transform. In *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR 2009)*, pages 1038–1045, Miami, FL, USA, 20-25 Jun 2009. doi:10.1109/CVPR.2009.5206693.

[27] B. Mali, A. Shrestha, A. Chapagain, et al. Challenges in the penetration of electric vehicles in developing countries with a focus on Nepal. *Renewable Energy Focus*, 40:1–12, 2022. doi:10.1016/j.ref.2021.11.003.

[28] A. Mohan and S. Poobal. Crack detection using image processing: A critical review and analysis. *Alexandria Engineering Journal*, 57(2):787–798, 2018. doi:10.1016/j.aej.2017.01.020.

[29] U. Nepal and H. Eslamiat. Comparing YOLOv3, YOLOv4 and YOLOv5 for autonomous landing spot detection in faulty UAVs. *Sensors*, 22(2):464, 2022. doi:10.3390/s22020464.

[30] H. Niigaki, J. Shimamura, and M. Morimoto. Circular object detection based on separability and uniformity of feature distributions using Bhattacharyya Coefficient. In *Proc. 21st Int. Conf. Pattern Recognition (ICPR2012)*, pages 2009–2012, Tsukuba, Japan, 11-15 Nov 2012. https://ieeexplore.ieee.org/abstract/document/6460553.

[31] T. Ojala, M. Pietikainen, and T. Maenpaa. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(7):971–987, 2002-07. doi:10.1109/TPAMI.2002.1017623.

[32] J. Pasquet, T. Desert, O. Bartoli, et al. Detection of manhole covers in high-resolution aerial images of urban areas by combining two methods. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 9(5):1802–1807, 2016. doi:10.1109/JSTARS.2015.2504401.

[33] S. Rao and N. Mitnala. Manhole_Detection. GitHub, Dec 2022. https://github.com/sh-r/Manhole_Detection. [Accessed: Dec, 2022].

[34] S. Rao, A. Quezada, S. Rodriguez, et al. Developing, analyzing, and evaluating vehicular lane keeping algorithms using electric vehicles. *Vehicles*, 4(4):1012–1041, 2022. doi:10.3390/vehicles4040055.

[35] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You Only Look Once: Unified, real-time object detection. In *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR 2016)*, pages 779–788, Las Vegas, NV, USA, 27-30 Jun 2016. doi:10.1109/CVPR.2016.91.

[36] J. Redmon and A. Farhadi. YOLO9000: Better, faster, stronger. In *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR 2017)*, pages 7263–7271, Honolulu, HI, USA, 21-26 Jul 2017. doi:10.1109/CVPR.2017.690.

[37] J. Redmon and A. Farhadi. YOLOv3: An incremental improvement. *arXiv*, 2018. arXiv:1804.02767v1. doi:10.48550/arXiv.1804.02767.

[38] A. Santos, J. Marcato Junior, J. Andrade Silva, et al. Storm-drain and manhole detection. Geomatics and Computer Vision Datasets. https://sites.google.com/view/geomatics-and-computer-vision/home/datasets#h.sen6zve8r3ra. [Accessed Jan, 2022].

[39] A. Santos, J. Marcato Junior, J. Andrade Silva, et al. Storm-drain and manhole detection using the RetinaNet method. *Sensors*, 20(16):4450, 2020. doi:10.3390/s20164450.

[40] P. Saxena. Increase Frame Per Second (FPS) rate in the custom object detection step by step. Towards Data Science, 3 Sep 2020. https://towardsdatascience.com/no-gpu-for-your-production-server-a20616bb04bd. [Accessed 17 Oct, 2022].

[41] M. Simonovsky. Ellipse detection using 1D Hough transform. In *MATLAB Central File Exchange*. 2022. [Accessed 16 Oct, 2022]. https://www.mathworks.com/matlabcentral/fileexchange/33970-ellipse-detection-using-1d-hough-transform.

[42] N. Tanaka and M. Mouri. A detection method of cracks and structural objects of the road surface image. In *Proc. IAPR Workshop on Machine Vision Applications*, pages 387–390, Tokyo, Japan, 28-30 Nov 2000. http://b2.cvl.iis.u-tokyo.ac.jp/mva/proceedings/CommemorativeDVD/2000/papers/2000387.pdf.

[43] R. Timofte and L. Gool. Multi-view manhole detection, recognition, and 3D localisation. In *Proc. IEEE Int. Conf. Computer Vision Workshops (ICCV Workshops)*, pages 188–195, Barcelona, Spain, 16 Jan 2011. Workshops. doi:10.1109/ICCVW.2011.6130242.

[44] H. Touvron, M. Cord, M. Douze, et al. Training data-efficient image transformers & distillation through attention. In M. Meila and T. Zhang, editors, *Proc. 38th Int. Conf. Machine Learning (ICML 2021)*, volume 139 of *Proceedings of Machine Learning Research*, pages 10347–10357, Virtual Only, 18-24 Jul 2021. PMLR. https://proceedings.mlr.press/v139/touvron21a.html.

[45] A. Vaswani, N. Shazeer, N. Parmar, et al. Attention is all you need. In *Advances in Neural Information Processing Systems (NIPS 2017)*, volume 30. 2017. https://proceedings.neurips.cc/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html.

[46] C. Y. Wang, A. Bochkovskiy, and H. Y. M. Liao. YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. *arXiv*, 2022. arXiv:2207.02696v1. doi:10.48550/arXiv.2207.02696.

[47] C.-Y. Wang, H.-Y. M Liao, Y.-H. Wu, et al. CSPNet: A new backbone that can enhance learning capability of CNN. In *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW 2020)*, pages 1571–1580, Seattle, WA, USA, 14-19 Jun 2020. doi:10.1109/CVPRW50498.2020.00203.

[48] C.-Y. Wang, H.-Y. M. Liao, I.-H. Yeh, et al. CSPNet: A new backbone that can enhance learning capability of CNN. *arXiv*, 2019. arXiv:1911.11929v1. doi:10.48550/arXiv.1911.11929.

[49] Y. Xie and Q. Ji. A new efficient ellipse detection method. In *Proc. 2002 Int. Conf. Pattern Recognition (ICPR 2002)*, volume 2, pages 957–960, Quebec City, QC, Canada, 11-15 Aug 2002. doi:10.1109/ICPR.2002.1048464.

[50] S. Yan. *Manhole Cover Detection from Natural Images*. PhD thesis, University of Dublin, Trinity College, Sep 2020. [Accessed 17 Oct, 2022]. https://www.scss.tcd.ie/publications/theses/diss/2020/TCD-SCSS-DISSERTATION-2020-111.pdf.

[51] Z. Yang, Y. Liu, L. Liu, X. Tang, J. Xie, and X. Gao. Detecting small objects in urban settings using SlimNet model. *IEEE Transactions on Geoscience and Remote Sensing*, 57(11):8445–8457, 2019. doi:10.1109/TGRS.2019.2921111.

[52] Y. Yu, H. Guan, and Z. Ji. Automated detection of urban road manhole covers using mobile laser scanning data. *IEEE Transactions on Intelligent Transportation Systems*, 16(6):3258–3269, 2015. doi:10.1109/TITS.2015.2413812.

Identifying Selected Diseases of Leaves using Deep Learning and Transfer Learning Models

Afsana Mimi[1], Sayeda Fatema Tuj Zohura[1], Muhammad Ibrahim[2*],
Riddho Ridwanul Haque[1], Omar Farrok[3], Taskeed Jabid[1], Md Sawkat Ali[1]

[1]*Department of Computer Science and Engineering, East West University, Dhaka, Bangladesh*
[2]*Department of Computer Science and Engineering, University of Dhaka, Dhaka, Bangladesh*
[3]*Ahsanullah University of Science and Technology, Dhaka, Bangladesh*

alim@ewubd.edu, afsana.mimi0304@gmail.com, syedafatema58@gmail.com,
**ibrahim313@du.ac.bd (corresponding author), riddhohaque@gmail.com, omar.eee@aust.edu,*
taskeed@ewubd.edu, alim@ewubd.edu

Abstract. Leaf diseases may harm plants in different ways, often causing reduced productivity and, at times, lethal consequences. Detecting such diseases in a timely manner can help plant owners take effective remedial measures. Deficiencies of vital elements such as nitrogen, microbial infections and other similar disorders can often have visible effects, such as the yellowing of leaves in *Catharanthus roseus* (bright eyes) and scorched leaves in *Fragaria* × *ananassa* (strawberry) plants. In this work, we explore approaches to use computer vision techniques to help plant owners identify such leaf disorders in their plants automatically and conveniently. This research designs three machine learning systems, namely a vanilla CNN model, a CNN-SVM hybrid model, and a MobileNetV2-based transfer learning model that detect yellowed and scorched leaves in *Catharanthus roseus* and strawberry plants, respectively, using images captured by mobile phones. In our experiments, the models yield a very promising accuracy on a dataset having around 4000 images. Of the three models, the transfer learning-based one demonstrates the highest accuracy (97.35% on test set) in our experiments. Furthermore, an Android application is developed that uses this model to allow end-users to conveniently monitor the condition of their plants in real time.

Keywords: convolutional neural network, transfer learning, leaf disease detection, image classification

1. Introduction

Visually observing the condition of the leaves of a plant is a good way to monitor its health and well-being. Yellowing and scorching of leaves are often symptoms of serious underlying conditions such as the deficiency of vital elements like nitrogen, surplus of chloride particles [14], or microbial infections caused by viruses, bacteria and fungi [25]. When such changes occur, it becomes necessary to identify and address the underlying issues as soon as possible. Microbial infections may often be contagious [29] and hence an early detection and removal of infected plants and leaves is crucial. Unless infected plants and leaves are identified and isolated in a timely manner, the infections may spread through a large number of plants in the vicinity, leading to drastically reduced production and having dire financial consequences [28].

Scientific evidence suggests that *C. roseus* leaves can be used for medicinal purposes [20], while the ubiquitous demand for strawberry as a fruit has led to its widespread industrial production [5]. Thus, both these plants are commonly cultivated in gardens and plantations, quite often in large amounts. Hence, manually monitoring these leaves for possible diseases is not a trivial task for gardeners and farmers. Automated tools can help in continuous monitoring of the health of these plants and in giving warning their users about signs of malnutrition and infectious diseases. The proposed approach takes a step in this direction and identifies signs of yellowing in the leaves of *C. roseus* and scorching in Strawberry plants. Our proposed system leverages machine learning techniques to detect signs of disease in plant leaves from their images.

1.1. Motivation and contribution

Machine learning is increasingly being applied to sectors like economy [19], social well-being [23], and agriculture [3]. Developing machine learning, in particular, deep learning models to identify leaf defects has been an area of interest in computer vision research for many years [8]. Despite this, to the best of our knowledge, there remains a lack of computer vision approaches that specifically focus on the detection of leaf damage in Bright Eyes and Strawberry plants. With a specific focus on detecting signs of yellowing and scorching in the leaves of these plants, we design, train, validate and compare the efficacy of multiple deep learning approaches for classifying plant leaves based on their images. The three approaches that we investigate are based on: (1) Convolutional Neural Network (CNN), (2) a hybrid architecture that combines CNN and Support Vector Machine (SVM), and (3) a transfer learning-based approach in which a MobileNetV2 model [13] pre-trained on the ImageNet Large Scale Visual Recognition Challenge 2012 (ILSVRC2012) dataset [17]. The experimental evaluation conducted on a real-world dataset consisting of both pre-existing and self-taken images shows the transfer-learning model to be the most effective among the three. The model is then loaded into an android application that facilitates easy offline access to plant owners around the world.

In this paper, a transfer-learning based approach is exploited for automated detection of yellow leaves in *Catharanthus roseus* (bright eyes) plants and scorched leaves in *Fragaria*$\times$*ananassa* (strawberry) plants from camera images. The key contributions of this paper are as follows:

- A novel dataset consisting of 3804 images is developed, 2239 of which are manually captured by us.
- Three approaches are designed to identify yellow and scorched leaves in Bright Eyes and Strawberry plants respectively. The approaches are: a vanilla CNN model, a CNN-SVM hybrid model, and a MobileNetV2-based transfer learning model. We also examine the effect of varying the number of training epochs on the accuracy of each approach.

- An Android application is developed that facilitates gardeners and farmers to easily utilize the transfer learning model for identification of unhealthy leaves in their plants.

Of the three models, the transfer learning-based one demonstrates the highest accuracy in our experiments which is 97.35% accuracy on test dataset.

The remainder of this paper is organized as follows. In Section 2 we provide a review of the relevant existing literature. Section 3 provides an overview of the proposed approach and describes the dataset and models. Section 4 describes the experimental results. Section 5 concludes the paper with directions for future research.

2. Related works

Use of deep learning models for detecting plant diseases has been a major focus of computer vision research over the years [3]. Sladojevic et al. [24] propose an approach based on CNN for detecting a variety of plant diseases. Das et al. [9] employ SVM to recognize leaf diseases in plants. Regarding hybrid architectures, CNN and SVM have been used in tandem by Ahlawat et al. [2] to identify handwritten characters.

In parallel with such generalized approaches, other techniques that are tailored to suit applications on particular species of plants are also studied in the literature. Kurtulmucs et al. [16] use three deep learning architectures, namely AlexNet [15], GoogLeNet [26] and ResNet [12], to classify sunflower seed images. Chen et al. [7] successfully use deep learning models to identify diseases in rice plants with satisfactory accuracy. Mokhtar et al. [18] compare different kernel functions in SVM for detecting tomato leaf diseases. Zhong et al. [30] use deep CNN for detecting leaf diseases in apple trees. Amara et al. [4] use CNN for detecting leaf diseases in banana trees.

Often, pre-processing the images before the training phase is vital for increasing the accuracy of machine learning models. Extracting features from datasets before training and testing models for image classification has been shown to be useful for improving the accuracy of the trained models. For example, Tiwari et al. [27], improve the accuracy of logistic regression model by using transfer learning-based feature extraction for potato disease classification. A summary of such specialized approaches which are designed to identify leaf diseases in specific types of plants is given in Table 1.

As Table 1 shows, deep learning models trained specifically to detect diseases in certain types of plants have shown relatively high levels of accuracy. Despite this, we think that the benefits of deep learning and transfer learning models are not fully harnessed in the existing literature. This research aims to bridge this gap by designing and evaluating several such models to identify yellow and scorched leaves in Bright Eyes and Strawberry plants respectively.

Tab. 1. Existing approaches to identifying leaf diseases in different types of plants.

Author	Algorithm	Plant Name	Accuracy
Kurtulmucs et al. [16]	DCNN	Sunflower	95%
Chen et al. [7]	Deep transfer learning	Rice	98.63%
Mokhtar et al. [18]	SVM	Tomato	99.83%
Tiwari et al. [27]	Transfer learning	Potato	97.8%
Zhong et al [30]	DCNN	Apple	93.71%
Amara et al. [4]	CNN	Banana	92%

3. Learning models

As with most typical machine learning-based prediction systems, our employed models consist, broadly, of two phases: (1) dataset preparation and processing, and (2) training and testing the models. This section discusses these details.

We prepare the dataset by first acquiring images of Bright Eyes and Strawberry leaves, by manually taking photographs using mobile phone cameras (2239 images) and from publicly available online sources (1565 images). Some sample images are shown in Figure 1. The images are resized and then compiled into a single dataset. Afterwards, the images are partitioned into training and test sets. Then, three CNN-based models are trained on this dataset: a vanilla CNN, a hybrid of CNN and SVM, and a transfer learning model, specifically, the MobileNetV2 model pre-trained on ILSVRC2012 dataset. All models are trained using the images in the training set and used to classify images in the test sets. The flowchart of the data preparation process along with learning modules is shown in Figure 2.

3.1. Data collection and preparation

As mentioned above, the dataset used in the experiments consists of 3804 images of *C. roseus* and strawberry leaves, of which 2239 are captured by us, and the rest 1565

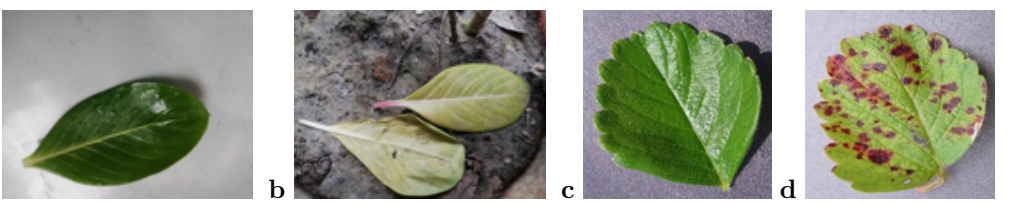


Fig. 1. Sample images of leaves from the dataset: (**a**) green *C. roseus*; (**b**) yellow *C. roseus*; (**c**) healthy strawberry; (**d**) scorched strawberry.

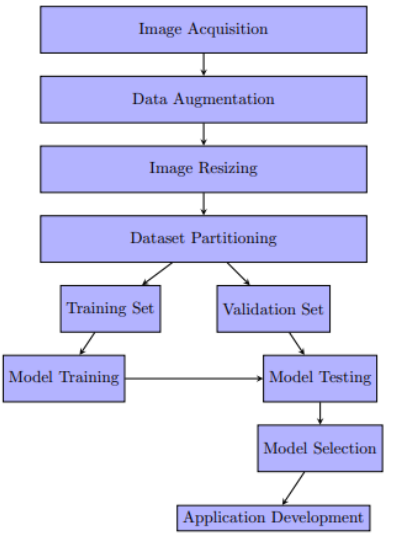


Fig. 2. Flow chart of the proposed methodology.

images are selectively chosen from the Internet. The images are then manually labeled by us into four types – green and yellow *C. roseus* leaves, and healthy and scorched strawberry leaves. Table 2 shows the number of images in each type of the dataset.

Data Augmentation

Since large training datasets help improve the accuracy of deep neural networks, we expand the size of the dataset using simple data augmentation techniques. Aside from increasing the number of training instances, data augmentation helps to diversify the dataset thereby allowing the trained model to be more robust to unnecessary variations and perturbations. All the images are re-scaled to 224×224 pixels before training using the ImageDataGenerator API provided in the Keras package [10]. These raw images are sheared, zoomed in, rotated and/or horizontally flipped. To do this, for every image of

Tab. 2. Statistics of different types of leaves in the dataset.

Image Type	Class Label	Number of Images
C. roseus Green	C0	1053
C. roseus Yellow	C1	1186
Strawberry healthy leaf	C2	456
Strawberry scorched leaf	C3	1109

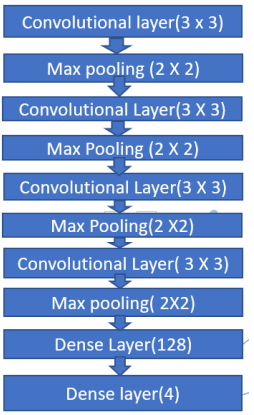


Fig. 3. Structure of employed vanilla CNN model (Model 1).

the dataset, we create three real-time tensor images using Keras platform: `shear_range`, `zoom_range`, `horizontal_flip`. However, the actual dataset does not contain these sheared, zoomed, and flipped images, rather we create those augmented images in real-time just before training.

All the images have three channels denoting red, green, and blue intensity levels of each of their pixels. Thus the total size of the dataset stands at 3804 images. The dataset is split into the training and test sets in 90-10 percentages (3426-378 images) respectively. We note here that in this paper we do not perform any hyper-parameter tuning using a separate validation dataset (which is left for future work). However, the platform we use for implementing the algorithms, namely Keras, uses the term *validation set* to indicate the test set. Therefore, throughout the paper we use the two terms *validation* and *test* interchangeably as there is no difference between these two terms in this paper.

3.2. Employed models

As mentioned earlier, in this investigation we utilize three machine learning models which are described below.

3.2.1. Model 1: CNN

The structure of the employed CNN model is illustrated in Figure 3 where a 3×3 convolutional layer followed by a 2×2 max-pooling layer are used. In total, four 3×3 convolutional layers and four 2×2 max-pooling layers are used. The popular activation function ReLU is used where a neuron is only activated when the output is greater than zero, so it does not activate all the neurons simultaneously. ReLU is popular because it

reduces the chance of facing the vanishing gradient problem, and often achieves better performance. Mathematically, ReLU is defined as $y = \max(0, x)$.

At the later stage, a hidden dense layer of 128 neurons is used with ReLU activation. Finally, as loss function the SoftMax activation along with the categorical cross entropy is utilized at the dense layer because it is able to re-scale the output. The formula for SoftMax activation function is

$$\text{SoftMax}(i) = \frac{e^i}{\sum_j e^j},$$

where j stands for the total number of neurons in the last layer. The model is compiled using a categorical cross-entropy loss function, to classify among multiple classes. The Adam optimizer is used to reduce the losses by its stochastic optimization approach.

The pseudo-code of the training phase of CNN is described below in Algorithm 1. The data preparation phase includes resizing the input image in 224×224 pixels and applying data augmentation to the input data.

Algorithm 1: CNN

```
begin
    for i ← 1 to 4 do
        pass the augmented and resized data into convolutional layer;
        pass the output of the last convolutional layer into max pooling layer;
    end
    pass the output of the previous layers into a flatten layer;
    pass the output of the flatten layer into a hidden later with 128 neurons;
    pass the output of the hidden layer using SoftMax activation function;
    calculate the loss using the categorical_crossentropy function;
end
```

Here is how it works: every input image is passed through four slices of convolutional and max pooling layers. Each slice consists of a 3×3 convolutional layer and a 2×2 max pooling layer. The output of these slices is passed through a flatten layer which converts the 2D array into a 1D array. After that the array is passed through a hidden layer consisting of 128 neurons. Finally, the output is passed through the final dense layer with 4 neurons.

3.2.2. Model 2: Hybrid of CNN and SVM

A hybrid of model consisting of an SVM with a mixture of a deep neural network is developed. Although there are some research on the hybrid CNN–SVM learning model in the existing literature, this research is still in an early phase, so it can be further improved by fine tuning its structure and parameters [21]. The features of input images are extracted using a CNN model and then fed into an SVM classifier in the last layer as

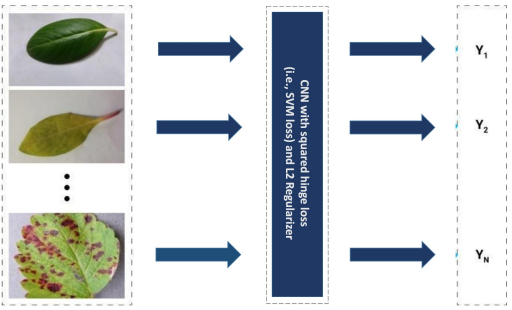


Fig. 4. Structure of hybrid CNN-SVM model (Model 2).

shown in Figure 4. The algorithm of the CNN-SVM training is described in Algorithm 2. The data preparation phase includes resizing the input image in 224×224 pixels and applying data augmentation to the input data.

Algorithm 2: Hybrid CNN-SVM

```
begin
    for i ← 1 to 4 do
        pass the augmented and resized data into convolutional layer;
        pass the output of the last convolutional layer into max pooling layer;
    end
    pass the output of the previous layers into a flatten layer;
    pass the output of the flatten layer into a hidden later with 128 neurons;
    pass the output of the hidden layer using SoftMax activation function;
    calculate the loss using the squared_hinge function (i.e., SVM loss) function
      with L2 regularizer;
end
```

Here is how it works: the images are passed through four slices like the previous CNN model. The output of these four slices are passed through a dense layer of 128 neurons. In this layer the ReLU is used as an activation function. In the last layer SoftMax activation function is applied.

The squared hinge loss function – also known as the SVM loss – is used, which enables us to draw a fine-tuned decision boundary between the classes. The formula of the squared hinge function is

$$\mathrm{L2}(y_1, y_2) = \sum_{i=1}^{n} \{\max(0, 1 - y_{1i} \cdot y_{2i})^2\} .$$

3.2.3. Model 3: Transfer Learning (MobileNetV2)

Transfer learning is a method through which the knowledge gained by training a model in one application area is utilized for classifying data in a different but related domain. The advantages of using pre-trained models include reduced training time and transfer of domain knowledge in terms of learnt weights of the network. Transfer learning models are particularly useful in scenarios where the target application contains a limited amount of training data. As a popular approach employed in various fields of data science, transfer learning has also been found to be effective in automated plant disease detection [30].

Our employed transfer learning model is based on the MobileNetV2 architecture [13]. The model is pre-trained on the ImageNet (ILSVRC-2012-CLS) dataset. The MobileNetV2 architecture is chosen due to its suitability for usage in mobile devices. The architecture minimizes the number of mathematical operations required, thus lessening the requirement of computational power. It uses Depthwise Separable Convolutions which makes it more efficient in comparison to other neural network architectures [22]. Thus MobileNetV2 is used to create a base model, and then a convolutional layer, dropout layer, global average pooling layer, and a dense layer are added on top of this model.

In the first layer, the pre-trained MobileNetV2 architecture model is invoked. A 2D convolution layer is then added on top of it. In the convolutional layer ReLU is used as a non-linear activation function. A dropout layer is then added in the model with a dropout rate of 0.2. On top of that was a 2D global average pooling layer which is used in lieu of a fully-connected layer. The final layer is a dense layer consisting of 4 neurons and a softmax function used for the activation. The dense layer produced the final output. We use the implementation of Keras library.

Pseudo-code of the transfer learning-based model is described below in Algorithm 3. The data preparation phase includes resizing the input image in 224×224 pixels and applying data augmentation to the input data.

Algorithm 3: Transfer learning (MobileNetV2) based model

```
begin
    load the MobileNetV2 as a base model and execute the top layer;
    apply convolutional layer to the output of the top layer;
    drop 20% of neurons by applying a dropout layer;
    pass the output of the last  layer into a GlobalAveragePooling2D layer;
    pass the output of the global average pooling layer using SoftMax activation function;
end
```

The intermediate layer of MobileNetV2 is used for feature extraction. After that a classifier is added on top of it. The model consists of a 3×3 convolutional layer,

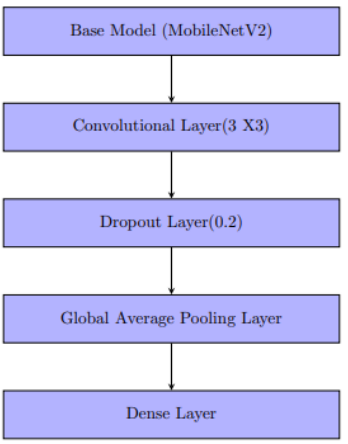


Fig. 5. Architecture of the employed transfer learning model (Model 3).

a dropout layer, a global average pooling layer and a dense layer consisting of 4 neurons. Adam optimizer and categorical-cross-entropy loss are applied while compiling this model.

3.3. Android application for leaf disease detection

After training and testing the transfer learning model, an Android application named "Go Greener" is implemented and deployed which allows users to easily use the system in their day-to-day gardening activities. The trained model is loaded into the mobile application using the Tensorflow Lite model. The application is developed using JAVA 15.0.1. The basic user interface of the application is shown in Figure 6.

The classification of the leaf's image is shown along with a confidence score. To detect the yellow leaves in *Catharanthus roseus* (bright eyes) plants and scorched leaves in *Fragaria* × *ananassa* (strawberry) plants from the "Go Greener" application, the user will simply have to open the app and place the phone on the plant leaf. The app shall then redirect the user to an interface where details on the status of the health of the leaf will be shown. Figure 7 pictorially presents the scenario.

4. Experimental results

All models are implemented in TensorFlow platform [1] which is an open source framework for developing machine learning projects in Python language. The number of training epochs varies from 1 to 10. A batch size of 64 is used. The training process is

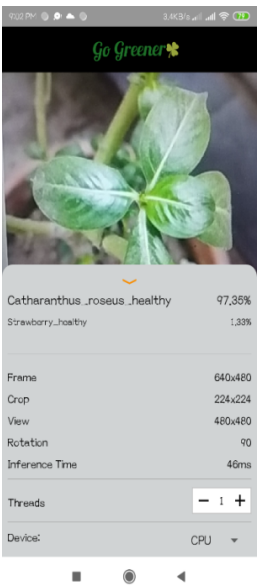


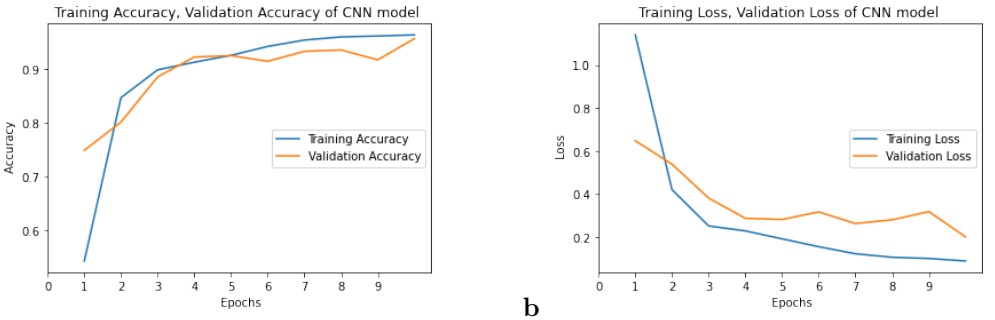
Fig. 6. Smartphone application layout.

Fig. 7. Use case diagram of the application.

accomplished on the Radeon Vega Mobile Gfx with CPU AMD Ryzen 5 3550H. All the three models are trained in the Windows 10 operating system, the CPU utilization is varying from 88% to 96% and the CPU clock speed is 2.19 GHz.

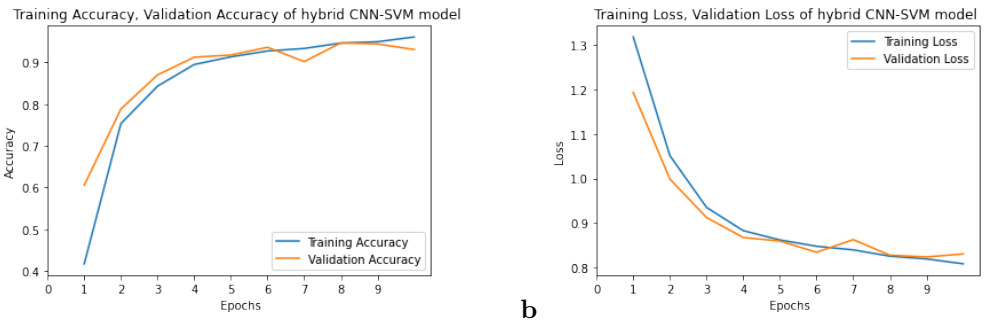
4.1. Model 1: CNN

Our CNN model gains a training accuracy of 96.47% and a validation accuracy of 95.77% after training for 10 epochs. The training phase takes 2 seconds per step. 10 training epochs are applied in the model. The influence of the number of epochs is displayed in Figure 8.



a **b**

Fig. 8. CNN model – evolution of quality metrics during training: (**a**) accuracy; (**b**) loss.



a **b**

Fig. 9. Hybrid CNN-SVM model – evolution of quality metrics during training: (**a**) accuracy; (**b**) loss.

4.2. Model 2: Hybrid CNN-SVM

The CNN-SVM model gives us 96.12% training accuracy and 93.12% validation accuracy for 10 epochs. The validation loss also significantly changes in this model, as shown in Figure 9 where the Squared Hinge loss function mostly reduces the large errors and it gives a computationally effective result.

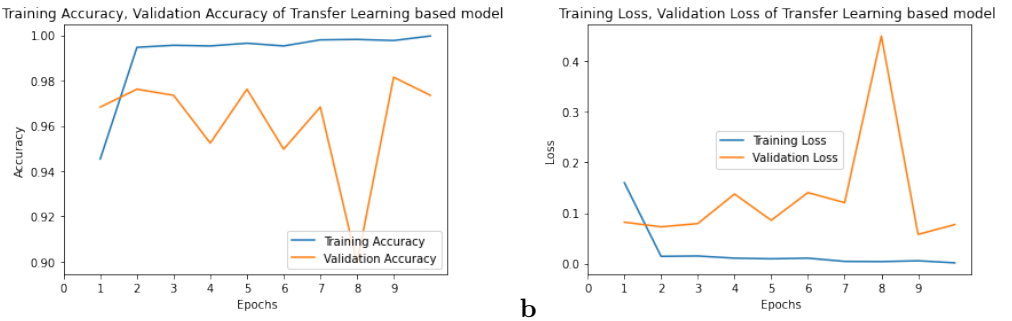
4.3. Model 3: Transfer Learning (MobileNet)

The employed transfer learning approach based on the MobileNetV2 architecture, outperforms the previous models. A training accuracy of 99.97% and validation accuracy of 97.35% are obtained using this model, making it the most effective among all three models. Table 3 describes the configuration of this model.

Figure 10 describes the changes in training and validation accuracy and loss with respect to the number of epochs for the transfer learning approach.

Tab. 3. Configuration of employed transfer learning-based model.

Parameter	Value
Convolutional Layer	1 (filter size: 3×3)
Global Average Pooling Layer	1 (filter size: 2×2)
Activation Function (Dense)	Softmax
Batch Size	64
Loss Function	`categorical_crossentropy`
Dropout	0.2
Number of Epochs	10



a **b**

Fig. 10. Transfer learning model – evolution of quality metrics during training: (**a**) accuracy; (**b**) loss.

4.4. Comparison among all three models

The training time taken by the vanilla CNN, hybrid CNN-SVM and the MobileNetV2-based transfer learning approach are recorded to be 15, 17 and 22 minutes, respectively. The resulting training and test accuracy metrics acquired using each model are compared in Table 4 and visualized in Figure 11.

All three investigated models perform impressively, achieving test accuracy of over 90%. However, the pre-trained transfer learning model outperforms the other two as can be seen in Figure 11a. Figure 11b shows how the training and test accuracy of each model varies with the number of training epochs.

Tab. 4. Accuracy comparison between the three models.

Algorithm	CNN	SVM	Transfer Learning
Training accuracy	96.47%	96.12%	99.97%
Test accuracy	95.77%	93.12%	97.35%

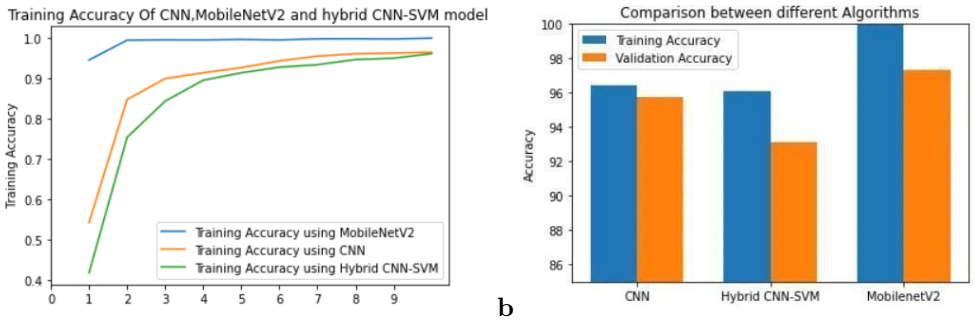


Fig. 11. Comparisons across three models, CNN, hybrid CNN-SVM, and transfer learning: (**a**) evolution of training accuracy versus training epochs; (**b**) final training and validation accuracies.

Based on precision, recall, F1-score, and accuracy, the performances of each model have been measured. Precision refers to the number of true positives divided by the total number of positive predictions. Recall is the number of true positives which are detected. F1-score is the balance between the precision and the recall which is computed by the following formula:

$$\text{F1-score} = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} .$$

These metrics are shown in Table 5.

If N is the number of classes to predict, the confusion matrix is an $N \times N$ matrix which is used to evaluate the performance of the classification of a model. As shown in Figure 12, using this matrix we can compare the actual target values with the predicted ones. In the case of the CNN model (Figure 12a), it is found that the rate of true positive value is high. It can be concluded that the rate of wrong predictions of the CNN model is quite acceptably low. In Figure 12b it is found that the SVM-CNN model results in good performance though it has a lower accuracy than the CNN model. The confusion matrix for MobileNetV2 is shown in Figure 12c where it is found that the transfer learning model gives us wrong predictions 10 times. All of these three models work satisfactorily as the true positive value is high for all the models.

Tab. 5. Precision, recall and F1-score measures of each model.

Algorithm	CNN	SVM	Transfer Learning
Precision	96.00%	95.00%	97.00%
Recall	96.00%	93.00%	97.00%
F1-score	96.00%	93.00%	97.00%

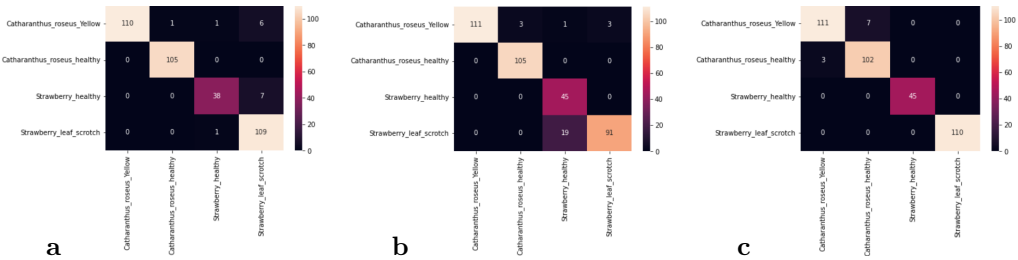


Fig. 12. Confusion matrices for (**a**) CNN model, (**b**) SVM model, (**c**) transfer learning model.

5. Conclusion

Computer vision is increasingly proving itself to be very effective to enhance the quality and production of plants. In this paper, a MobileNetV2-based transfer learning model is employed along with two other CNN-based models to classify the green and yellow leaves in *Catharanthus roseus* (bright eyes) leaves and also to classify healthy and scorched leaves of *Fragaria*×*ananassa* (strawberry) plants. Satisfactory accuracy has been achieved in the experiments. Once trained, the models can be loaded and used in smartphone applications, which can be used by end-users to classify images of leaves in real time using their mobile phone cameras. The model is expected to be helpful for cultivators in detecting diseases within bright eyes and strawberry plants. Future research in this field may generate several interesting directions. The deployed transfer learning approach can be tweaked further to improve performance and can be trained to identify diseases in other plants with even greater economic significance. Other transfer learning approaches can be considered including ResNet, GoogLeNet and EfficientNet [11]. Since the developed dataset is somewhat imbalanced in terms of class distribution, re-sampling techniques [6] can be applied to improve the accuracy.

References

[1] M. Abadi, A. Agarwal, P. Barham, et al. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. `https://www.tensorflow.org/`. Software available from tensorflow.org.

[2] S. Ahlawat and A. Choudhary. Hybrid CNN-SVM classifier for handwritten digit recognition. *Procedia Computer Science*, 167:2554–2560, 2020. doi:10.1016/j.procs.2020.03.309.

[3] S. I. Ahmed, M. Ibrahim, Md. Nadim, et al. MangoLeafBD: A comprehensive image dataset to classify diseased and healthy mango leaves. *Data in Brief*, 47:108941, 2023. doi:10.1016/j.dib.2023.108941.

[4] J. Amara, B. Bouaziz, and A. Algergawy. A deep learning-based approach for banana leaf diseases classification. In B. Mitschang et al., editors, *Datenbanksysteme für Business, Technologie und Web (BTW 2017) – Workshopband*, pages 79–88, Bonn, 2017. Gesellschaft für Informatik e.V. `https://dl.gi.de/handle/20.500.12116/944`.

[5] L. E. C. Antunes and N. A. Peres. Strawberry production in Brazil and South America. *International Journal of Fruit Science*, 13(1-2):156–161, 2013. doi:10.1080/15538362.2012.698147.

[6] L. E. C. Antunes and N. A. Peres. Sampling non-relevant documents of training sets for learning-to-rank algorithms. *International Journal of Machine Learning and Computing*, 10(3):406–415, 2020. doi:10.18178/ijmlc.2020.10.3.950.

[7] J. Chen, D. Zhang, Y. A. Nanehkaran, and D. Li. Detection of rice plant diseases based on deep transfer learning. *Journal of the Science of Food and Agriculture*, 100(7):3246–3256, 2020. doi:10.1002/jsfa.10365.

[8] S. S. Chouhan, U. P. Singh, and S. Jain. Applications of computer vision in plant pathology: A survey. *Archives of Computational Methods in Engineering*, 27(2):611–632, 2020. doi:10.1007/s11831-019-09324-0.

[9] D. Das, M. Singh, S. S. Mohanty, and S. Chakravarty. Leaf disease detection using Support Vector Machine. In *Proc. 2020 Int. Conf. Communication and Signal Processing (ICCSP)*, pages 1036–1040, Chennai, India, 28-30 Jul 2020. IEEE. doi:10.1109/ICCSP48568.2020.9182128.

[10] Keras Special Interest Group. Keras. simple. flexible. powerful. `https://keras.io`.

[11] Md. M. Hasana, M. Ibrahim, and Md. S. Ali. Speeding up EfficientNet: Selecting update blocks of convolutional neural networks using genetic algorithm in transfer learning. *arXiv*, 2023. arXiv:2303.00261v1. doi:10.48550/arXiv.2303.00261.

[12] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR 2016)*, pages 770–778, Las Vegas, NV, USA, 27-30 Jun 2016. doi:10.1109/CVPR.2016.90.

[13] A. G. Howard, M. Zhu, B. Chen, et al. MobileNets: Efficient convolutional neural networks for mobile vision applications. *arXiv*, 2017. arXiv:1704.04861v1. doi:10.48550/arXiv.1704.04861.

[14] A. J. Keutgen and E. Pawelzik. Impacts of NaCl stress on plant growth and mineral nutrient assimilation in two cultivars of strawberry. *Environmental and Experimental Botany*, 65(2-3):170–176, 2009. doi:10.1016/j.envexpbot.2008.08.002.

[15] A. Krizhevsky, I. Sutskever, and G. E. Hinton. ImageNet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems 25 – Proc. 25th Conf. NIPS 2012*, volume 25, pages 1097–1105, Lake Tahoe, NV, USA, 3-6 Dec 2012. Curran Associates, Inc. `https://proceedings.neurips.cc/paper_files/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf`.

[16] F. Kurtulmuş. Identification of sunflower seeds with deep convolutional neural networks. *Journal of Food Measurement and Characterization*, 15:1024–1033, 2021. doi:10.1007/s11694-020-00707-7.

[17] Large Scale Visual Recognition Challenge 2012 (ILSVRC2012). Stanford Vision Lab, 2012. `http://www.image-net.org/challenges/LSVRC/2012/results`.

[18] U. Mokhtar, N. El Bendary, A. E. Hassenian, et al. SVM-based detection of tomato leaves diseases. In D. Filev, J. Jabłkowski, J. Kacprzyk, et al., editors, *Intelligent Systems'2014. Advances in Intelligent Systems and Computing. Proc. 7th IEEE Int. Conf. Intelligent Systems (IS'2014)*, volume 323 of *Advances in Intelligent Systems and Computing*, pages 641–652. Springer, Warsaw, Poland, 24-26 Sep 2015. doi:10.1007/978-3-319-11310-4_55.

[19] T. Muhammad, A. B. Aftab, Md. Ahsan, et al. Transformer-based deep learning model for stock price prediction: A case study on Bangladesh stock market. *International Journal of Computational Intelligence and Applications*, 22(1):2350013, 2023. doi:10.1142/S146902682350013X.

[20] S. Nammi, M. K. Boini, S. D. Lodagala, and R. B. S. Behara. The juice of fresh leaves of Catharanthus roseus Linn. reduces blood glucose in normal and alloxan diabetic rabbits. *BMC Complementary and Alternative Medicine*, 3(1):4, 2003. doi:10.1186/1472-6882-3-4.

[21] X.-X. Niu and C. Y. Suen. A novel hybrid CNN-SVM classifier for recognizing handwritten digits. *Pattern Recognition*, 45(4):1318–1325, 2012. doi:10.1016/j.patcog.2011.09.021.

[22] M. Sandler, A. Howard, M. Zhu, et al. MobileNetV2: Inverted residuals and linear bottlenecks. In *Proc. 2018 IEEE/CVF Conf. Computer Vision and Pattern Recognitioneedings (CVPR 2018)*, pages 4510–4520, Salt Lake City, UT, USA, 18-23 Jun 2018. doi:10.1109/CVPR.2018.00474.

[23] F. T. Shohan, A. U. Akash, M. Ibrahim, and M. S. Alam. Crime prediction using machine learning with a novel crime dataset. *arXiv*, 2022. doi:10.48550/arXiv.2211.01551.

[24] S. Sladojevic, M. Arsenovic, A. Anderla, et al. Deep neural networks based recognition of plant diseases by leaf image classification. *Computational Intelligence and Neuroscience*, 2016:3289801, 2016. doi:10.1155/2016/3289801.

[25] S. Sun, Z. Wang, Z. Zhang, et al. The roles of entomopathogenic fungal infection of viruliferous whiteflies in controlling tomato yellow leaf curl virus. *Biological Control*, 156:104552, 2021. doi:10.1016/j.biocontrol.2021.104552.

[26] C. Szegedy, W. Liu, Y. Jia, et al. Going deeper with convolutions. *arXiv*, 2014. arXiv:1409.4842v1. doi:10.48550/arXiv.1409.4842.

[27] D. Tiwari, M. Ashish, N. Gangwar, et al. Potato leaf diseases detection using deep learning. In *Proc. 2020 4th Int. Conf. Intelligent Computing and Control Systems (ICICCS 2020)*, pages 461–466, Madurai, India, 13-15 May 2020. IEEE. doi:10.1109/ICICCS48265.2020.9121067.

[28] M. Vurro, B. Bonciani, and G. Vannacci. Emerging infectious diseases of crop plants in developing countries: impact on agriculture and socio-economic consequences. *Food Security*, 2(2):113–132, 2010. doi:10.1007/s12571-010-0062-7.

[29] P. E. Waggoner. The aerial dispersal of the pathogens of plant disease. *Philosophical Transactions of the Royal Society B, Biological Sciences*, 302(1111):451–462, 1983. doi:10.1098/rstb.1983.0067.

[30] Y. Zhong and M. Zhao. Research on deep learning in apple leaf disease recognition. *Computers and Electronics in Agriculture*, 168:105146, 2020. doi:10.1016/j.compag.2019.105146.

Riesz-Laplace Wavelet Transform and PCNN Based Image Fusion

Shuifa Sun[1,2], Yongheng Tang[2,3], Zhoujunshen Mei[2], Min Yang[2],
Tinglong Tang[1,2], Yirong Wu[4]

[1]*Hubei Key Laboratory of Intelligent Vision Based Monitoring for Hydroelectric Engineering, China Three Gorges University, YiChang, China*

[2]*College of Computer and Information Technology, China Three Gorges University, YiChang, China*

[3]*College of Economics and Management, China Three Gorges University, YiChang, China*

[4]*Institute of Advanced Studies in Humanities and Social Sciences, Beijing Normal University, Zhuhai, China*

Corresponding author: Yirong Wu, yrwu@bnu.edu.cn

Abstract. Important information perceived by human vision comes from the low-level features of the image, which can be extracted by the Riesz transform. In this study, we propose a Riesz transform based approach to image fusion. The image to be fused is first decomposed using the Riesz transform. Then the image sequence obtained in the Riesz transform domain is subjected to the Laplacian wavelet transform based on the fractional Laplacian operators and the multi-harmonic splines. After Laplacian wavelet transform, the image representations have directional and multi-resolution characteristics. Finally, image fusion is performed, leveraging Riesz-Laplace wavelet analysis and the global coupling characteristics of pulse coupled neural network (PCNN). The proposed approach has been tested in several application scenarios, such as multi-focus imaging, medical imaging, remote sensing full-color imaging, and multi-spectral imaging. Compared with conventional methods, the proposed approach demonstrates superior performance on visual effects, contrast, clarity, and the overall efficiency.

Key words: image fusion, Riesz transform, polyharmonic spline, Laplacian wavelet, pulse coupled neural network, PCNN.

1. Introduction

Image fusion aims to integrate images obtained by different focusing positions or different kinds of sensors into one new image containing better description of the scene. Different focusing positions result in different sharpness for different areas of an image. In order to obtain clear images and more information, image fusion methods are usually adopted to solve this kind of multi-focus problem [9]. In addition to multi-focus problem, there are many other image fusion problems. With the development of new technology, there are more and more types of imaging sensors. Different sensors have different capabilities of acquiring different information. For example, for multi-band remote sensing images, full-color images and multi-spectral images can be collected by remote sensing satellite imaging equipment, in which full-color images have structure information with high spatial resolution, and multi-spectral images contain color information. Fused

images of these two kinds of images can be used to better describe actual ground information [21, 22]. At present, image fusion can be performed in either spatial domain or transform domain. Image fusion in spatial domain directly operates on the gray value of an image. This method cannot extract edge information, which results in low contrast for fusion results. Image fusion in transform domain attracts a lot of attention currently, including pyramid decomposition, wavelet transform and other multi-scale transforms. The pyramid-based image fusion method can retain the edge details of the original image and make the fusion image stable and anti-noise. But in the multi-scale structures of pyramid decomposition, the details of different resolutions are strongly correlated, so the fusion results are not satisfactory [18]. Wavelet transform is the most commonly used method in transform domain since it reduces the correlation between sub-band coefficients in terms of time-frequency local characteristics [2]. Leveraging these two methods, we develop an image fusion approach. In image fusion applications for either different focusing positions or different kinds of sensors, our proposed approach demonstrates superior performance on visual effects, contrast, clarity, and the overall efficiency.

2. Related works

Image can be fused at pixel-level, feature-level, or decision making-level. Most of studies focus on pixel-level and feature-level fusion [4, 17]. This structure information are often extracted in the transform domain instead of the spatial domain for image fusion since it is easier to separate this information by directly operating image gray values in the transform domain to obtain clearer and higher contrast images. Most of related studies in the transform domain focus on multi-resolution transform, such as pyramid decomposition, wavelet transform, super-wavelet transform, etc. [2, 18, 22]. Compared with pyramid-based decomposition, wavelet decomposition performs well in local frequency analysis, and reduces the correlation between sub-band coefficients [14]. The contourlet transform, one of the super-wavelets, has multi-directionality feature and can represent singular lines and surfaces in natural scenes better than wavelet transform [8]. Additionally, non-subsampled Contourlet Transform (NSCT) uses a sub-band separation method to avoid spectrum aliasing caused by up-down sampling process [21] but space and time complexity are increased. The common goal of this series of improvements is to better decompose images and gather related features to facilitate separation. In order to solve this problem, in this paper a fusion method of Riesz transform is proposed for unfused image, which can express the local structure information of an image well and maintain the consistency of low-level feature space [22]. Riesz transform has been widely used in computer vision and image processing, such as image quality evaluation based on image structure similarity after this transform [3, 10]. Because Riesz transform is isotropic and relatively easy to calculate, it is more efficient for detecting the edges of an image based on phase consistency in Riesz transform space than that in Hilbert transform space [11].

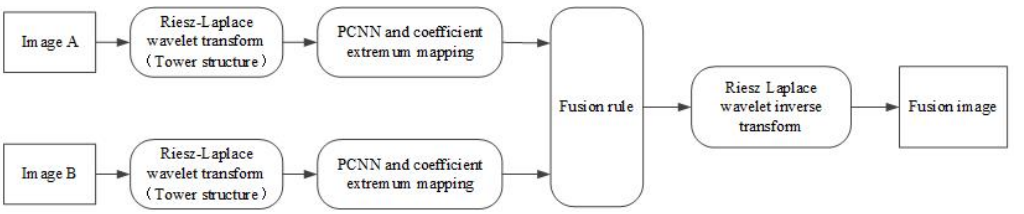


Fig. 1. Riesz-Laplace wavelet pyramid combined with PCNN image fusion process block diagram.

In this study, we propose a Riesz transform based image fusion approach by leveraging important characteristics of this transform, such as low-level feature preservation, locality and directionality. An image is first transformed into Riesz transform space. After this operation, the transformed images are further transformed into Laplacian wavelet transform space, so that the image has multi-resolution characteristics. The Laplacian wavelet transform not only reduces a large amount of redundant information in a single pyramid transform, but also has good spatial and frequency domain locality in the transform space, which can be used to well retain high-frequency information of the fused image [18]. Then, the unfused images of each layer are input into Pulse Coupled Neural Network (PCNN). PCNN is an improvement of the neuron model proposed by Eckhorn et al., which is based on the phenomenon of cerebral cortex sync pulse distribution in cats [12, 13]. It has the characteristics such as spatio-temporal summation, dynamic pulse release, vibration, and fluctuation caused by synchronous pulse release. It has been widely used in image denoising, enhancement, segmentation, edge detection, and fusion. In this paper, the multi-resolution image is processed by PCNN, and the neurons ignition frequency graph describing feature clustering is obtained. Then, based on the size of the ignition frequency graph, each scale of the original image is fused [13]. Finally, the results from PCNN are inversely transformed into the Riesz transform domain by Laplacian pyramid. The inverse transform of Riesz (here multiplied by the conjugate coefficient of Riesz transform) finally produces the fused image.

3. Riesz transformation principle

Since Gabor proposed analytic signal in 1946, Hilbert transform has been widely used in one-dimensional signal analysis. One-dimensional analytic signal is a complex signal composed of two parts. The first part is the real part, the original signal, and the second part is the imaginary part derived from Hilbert transform of original signal. Local amplitude and phase can be obtained based on one-dimensional analytic signal analysis. Before Riesz transform was developed, there have been many studies on two-dimensional extension of Hilbert transform. However, analytic signals obtained in these studies have the problem of information loss [11].

Riesz transform can be regarded as an extended Hilbert transform of one-dimensional signal in multi-dimensional space. It has similar properties to those of Hilbert transform. The representation of Hilbert transform kernel in frequency domain is as follows:

$$H(w) = -j \cdot \text{sign}(w) = -j\frac{w}{||w||}, \tag{1}$$

where w is the frequency of an one-dimensional signal. The transform kernel of the Riesz transform in frequency domain is a two-dimensional vector (R_x, R_y) containing two parts:

$$(R_x, R_y) = \left(\frac{w_x}{||w||}, \frac{w_y}{||w||}\right), \tag{2}$$

where w stands for for the two-dimensional vector $w = (w_x, w_y)$, $||w||$ is the modulo value of the vector $|w| = (w_x^2 + w_y^2)^{\frac{1}{2}}$.

When an image is defined as $f(x, y)$, any point on the image is defined as $P = (x, y)$, the Riesz transform kernel in space domain can be described by

$$(r_x, r_y) = \left(\frac{x}{2\pi\,||p||^3}, \frac{y}{2\pi\,||p||^3}\right). \tag{3}$$

Riesz transform of an image can be expressed by equation (4), where $*$ represents a convolution operation. The image in spatial domain is convolved with two Riesz transform kernels separately.

$$f_r(P) = \begin{pmatrix} f_{r_x} \\ f_{r_y} \end{pmatrix} = \begin{pmatrix} f(P) * r_x \\ f(P) * r_y \end{pmatrix}. \tag{4}$$

In the frequency domain, the image is first transformed by Fourier transform. Then, it is multiplied by the frequency domain Riesz transform kernel:

$$F_R(w) = \begin{pmatrix} F_{R_x} \\ F_{R_y} \end{pmatrix} = \begin{pmatrix} R_x \cdot F(w) \\ R_y \cdot F(w) \end{pmatrix}. \tag{5}$$

The second-order transform of the Riesz transform is expressed as follows:

$$\begin{cases} R_{xx}[f](P) = R_x[R_x[f](P)](P) \\ R_{xy}[f](P) = R_y[R_x[f](P)](P) \\ R_{yy}[f](P) = R_y[R_y[f](P)](P) \end{cases}. \tag{6}$$

The first-order feature map can well express the edge contour features of the image, and the second-order feature map can express more complex features, such as corner points. As shown in Fig. 2, R_x and R_{xx} highlight the horizontal contour of the image, R_y and R_{yy} the vertical contour, and R_{xy} the diagonal profile [14,17]. The higher-order Riesz transform is directional [8].

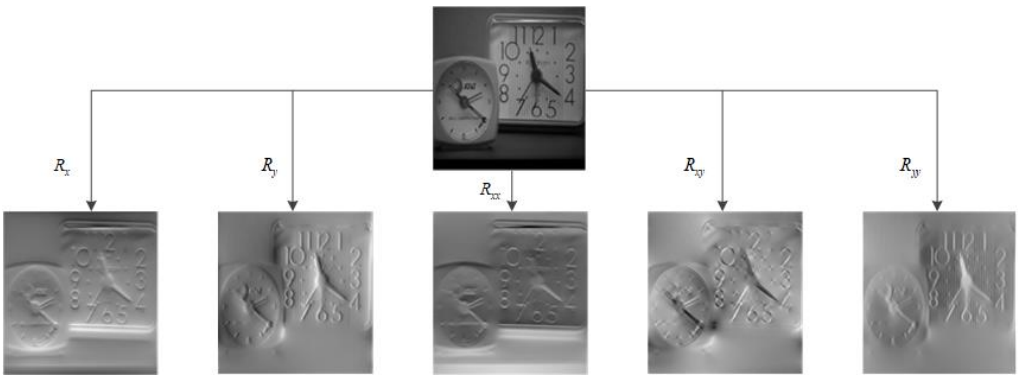


Fig. 2. First and second order Riesz transform image schematic diagram, the first row of images (source images) comes from the TestingImageDataset in the image fusion standard database [16], and the second row of images is the first-order feature map and second-order feature map after Riesz transformation.

4. Laplacian wavelet transform based on Riesz transform

Based on the Riesz transform, Laplacian spline wavelets are used for multiresolution decomposition. The fractional Laplacian operator $(-\Delta)^{\alpha}$, $\alpha \in R^{+}$, is an isotropic differential operator of order 2α. In Fourier domain it is defined as:

$$(-\Delta)^{\alpha} f(x) \xrightarrow{\text{FT}} ||w||^{2\alpha} \hat{f}(w)\,. \tag{7}$$

The multi-harmonic spline is a spline function related to the fractional Laplacian operator, and the γ-order multi-spline function $\varphi_{\gamma}(x)$ satisfies:

$$s(x) = \sum_{k \in Z^2} s[x] \varphi_{\gamma}(x-k)\,, \tag{8}$$

where $s[x] = s(x)|_{x=k}$ is the integer sample of $s(x)$ at k. The Fourier response of $\varphi_{\gamma}(x)$ is described as:

$$\varphi_{\gamma}(x) \xrightarrow{FT} \hat{\varphi}_{\gamma}(w) = \frac{||w||^{-\gamma}}{\sum_{k \in Z^2} ||w+2\pi k||^{-\gamma}}\,. \tag{9}$$

With the above concepts of fractional differential operators and multi-harmonic splines, a symmetric fractional order ($\gamma > \frac{1}{2}$) Laplacian spline wavelet can be defined as:

$$\psi(x) = (-\Delta)^{\frac{\gamma}{2}} \varphi_{2\gamma}(Dx)\,, \tag{10}$$

where $\varphi_{2\gamma}$ is the 2γ order multi-harmonic spline interpolation operator; the parameter of D is the down-sampling matrix. The basis function defined by $\psi(x)$ is:

$$\psi_{i,k}(x) = |det(D)|^{\frac{i}{2}}\, \psi(D^i x - D^{-1}k)\,, \tag{11}$$

where the parameters i, k are the scaling and translation coefficients, respectively. The Riesz transform is performed on the Laplacian wavelet base to obtain complex Riesz Laplacian wavelets of order $\gamma(\gamma > \frac{1}{2})$:

$$\psi'(x) = R\psi(x) = R(-\Delta)^{\frac{\gamma}{2}} \varphi_{2\gamma}(Dx)\,. \tag{12}$$

5. The fusion process

We observe that low-level features of an image are separated well by Riesz Laplacian wavelet transform. Additionally, the selection of proper fusion rules is also important in the whole process of image fusion. This paper uses the Pulse Coupled Neural Network to perform image fusion at each layer of the Riesz-Laplace wavelet transform pyramid.

Pulse Coupled Neural Network is a bionic model proposed by Eckhorn in the 1990s, and is based on synchronous pulses in the cerebral cortex of animals such as cats and monkeys. PCNN is a simplified neural network which imitates the principle of cat vision. It is a feedback network composed of several interconnected neurons and has the effect of synchronous pulse release [12, 13].

The traditional PCNN fusion rules are:

$$I^k_{F.l}(x,y) = \begin{cases} I^k_{A.l}(x,y) & T^k_{A.l}(x,y) \geq T^k_{B.l}(x,y) \\ I^k_{B.l}(x,y) & T^k_{A.l}(x,y) < T^k_{B.l}(x,y) \end{cases}, \tag{13}$$

where $I^k_{A.l}(x,y)$ and $I^k_{B.l}(x,y)$ are the wavelet coefficients of the source image A and the source image B, and $T^k_{A.l}(x,y)$ and $T^k_{B.l}(x,y)$ are the numbers of firings corresponding to the wavelet coefficients.

The steps of PCNN image fusion based on Riesz-Laplace wavelet transform are described as follows:

a) Riesz-Laplace wavelet transform is applied to the unfused image A and B to obtain the pyramid structures of the images. Laplacian wavelet transform based on Riesz transform has the coefficients of each layer are $I^k_{A.l}(x,y,i)$ and $I^k_{B.l}(x,y,i)$.
b) The wavelet coefficients of each layer obtained from a) are input into the PCNN network separately, and iterate $N_{\max}$ times to obtain the respective ignition maps of each layer, called also signature maps, denoted as $\mathrm{MF}_{A.i}$ and $\mathrm{MF}_{B.i}$.
c) The pixel points with a larger ignition time at each layer are taken as the fusion coefficients for image fusion.
d) The inverse Riesz Laplacian wavelet transform is finally performed on obtained coefficients to recover the fused image.

6. Experiment and result analysis

6.1. Experimental description

The quality of fused images can be evaluated subjectively and objectively. In order to objectively test the method of this paper, to verify the universality and accuracy of the algorithm in this paper, in this experiment multiple images in different scenes from the image fusion standard database for fusion testing are randomly selected, and compared with LP-PCNN [5] (its preliminary version can be found in [6]) and NCST-PCNN [23] algorithms. Among them, the selected image data comes from TestingImageDataset from [15] (mainly including multi-focus image fusion data), Harvard [7] (mainly including medical image fusion data), and three Gaofen data bases: GF-1, GF-2 and GF-3 from [1] (mainly including remote sensing image fusion data) in the image fusion standard database [16].

The experimental environment of this article is 11th Gen Intel(R) Core (TM) i7-11700 @ 2.50 GHz, memory 16 GB, 64-bit Win10 operating system, and Matlab R2018b programming environment [20] in which multi-spectral and full-color remote sensing images have multiple bands. For this kind of images, first, they are converted from RGB space to HIS space. The I component (I represents the I component of HIS image) containing the main structure information is fused with full-color remote sensing image using the method proposed in this paper. Then the new I component and the multispectral H and S components with spectral features are transformed back to the RGB space. The images to be fused in this paper have been initially registered. The results of image fusion are shown in Fig. 3.

6.2. Subjective evaluation

Subjective evaluation mainly depends on human visual effects. Three methods including Laplacian pyramid decomposition and restoration(, LP-PCNN [5] (also [6]), NCST-PCNN [23] and our method have been used to fuse three different types of images. For multi-focusing clock image, the decomposition of the Laplacian pyramid produces artifacts. NSCT-PCNN and our method perform well. For medical images, Laplacian decomposition produces artifacts, which is not good for medical diagnosis. The fusion effects of NSCT-PCNN and our method are similar. For remote sensing images, images fused by the Laplacian pyramid method are dark. The fusion effects of NSCT-PCNN and our method are superior to LP-PCNN.

In particular in Fig. 3, the images in the first row are the two images to be fused, and the images in the second row represent images generated by fusion of three different methods, where (1) represents Laplacian Pyramid decomposition and restoration using Pulse Coupled Neural Network fusion strategy (LP-PCNN) method. The result map generated by fusion (2) represents the result map generated by the fusion of non-downsampled

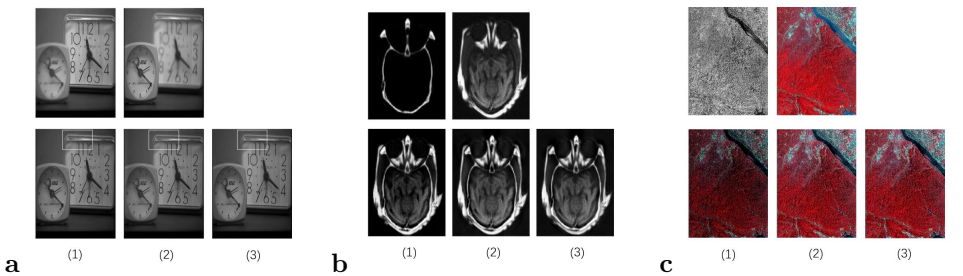


Fig. 3. Image fusion results. (**a**) The Clock fusion; (**b**) Medical CT, MRI fusion; (**c**) Remote sensing full-color and multi-spectral image fusion. Laplacian Pyramid decomposition and restoration using Pulse Coupled Neural Network fusion strategy (LP-PCNN) (1), non-downsampled contourlet transform decomposition and restoration (NSCT-PCNN) (2), and Riesz fractional Laplacian Pyramid decomposition and restoration (*Ours*) (3) fusion methods test for the above multi-scenario application. The white window is to focus on the details to illustrate the comparative advantages of the fusion results of this method and the fusion results of other methods. Taking the multi-focus clock image as an example in Fig. 3(a), the method of LP-PCNN will produce artifacts, while NSCT-PCNN and the method proposed in this paper perform well.

contourlet transform decomposition and restoration ($NCST\_PCNN$) method, and (3) represents the result map generated by the fusion method of this paper.

6.3. Objective evaluation

In order to evaluate the fusion results more comprehensively, objective indexes are used in this paper. Objective evaluation of the effect of image fusion is standard and deterministic. For this reason, in this paper the information entropy (IE), standard deviation (SD), and average gradient (AG) defined in the literature [19] were chosen to evaluate the fusion results objectively. This kind of evaluation index is to use the fused image itself or some feature statistics to measure the quality of the fused image, and then make an objective evaluation of the performance of the image fusion technology. The specific meanings of each evaluation index are as follows. 1. Information entropy is an important index to measure the information richness of an image. The larger the entropy value, the more information the image contains. 2. The larger the standard deviation, the more dispersed the gray level distribution of the image, the greater the contrast, and more information can be seen, the better the image quality; the smaller the standard deviation, the worse the image quality. The smaller the contrast of the image, the image is closer to an image with single and uniform tone, and not much information can be seen. 3. The average gradient can sensitively reflect the clarity of the image, and the larger the average gradient, the richer the information obtained from the source image of the fused image, and the better the fusion.

Information Entropy (IE)
Information entropy reflects the richness of image information.

$$\mathrm{IE} = -\sum_{i=0}^{L-1} P_i \ln P_i \,. \tag{14}$$

In the formula, P_i represents the ratio of the number of pixels with gray value i in the image to the total number of pixels in the image, which reflects the probability distribution of pixels with gray value i in the image. The larger the value of IE, the richer the information and the better the quality of the fused image.

Standard Deviation (SD)
The standard deviation indicates the degree of dispersion of the data. The larger the standard deviation, the more discrete the data, and the larger the standard deviation, the larger the contrast is reflected in the image.

$$\mathrm{SD} = \sqrt{\frac{1}{X \times Y} \sum_{x=1}^{X} \sum_{y=1}^{Y} (p(x,y) - \mu)^2} \,. \tag{15}$$

In the formula, μ represents the mean value of the gray value of the image, which reflects the average brightness of the image, and the standard deviation reflects the sharpness and contrast of the image. The higher the SD value, the better the contrast and the clearer the fusion result.

Average Gradient (AG)
Average Gradient (AG) reflect the clarity of the image, The larger the AG value, the clearer the fused image and the better the quality.

$$\begin{cases} AG_v = \frac{\sum_{x=0}^{X-1} \sum_{y=0}^{Y-2} |p(x+1,y) - p(x,y)|}{(X-1) \times Y} \\ AG_h = \frac{\sum_{x=0}^{X-2} \sum_{y=0}^{Y-2} |p(x,y+1) - p(x,y)|}{X \times (Y-1)} \end{cases} \quad AG = \sqrt{{AG_v}^2 + {AG_h}^2} \,. \tag{16}$$

In practical applications, time complexity is an important factor in decision making for algorithm selection. In this study, the efficiency of algorithms is also one of the criteria to evaluate the quality of algorithms. The statistical results are described in Table 1, in which the multi-spectral and full-color image results are the mean of three-channel results. The values of IE, SD, and AG are positively correlated with the fusion effects; the higher the values, the better image fusion effects. In multi-focus image fusion and CT/MRI image fusion, our method performs better than NSCT-PCNN method, especially in time complexity analysis. Compared with the Laplacian decomposition method, our method produces higher IE values, the image results are shown in Fig. 3(a) and Fig. 3(b). In multi-spectral and full-color image fusion, our method produces higher index values than other algorithms, the image results are shown in Fig. 3(c).

Tab. 1. Image fusion evaluation statistics.

Unfused Images	Fusion method	IE	SD	AG	Time [s]
Multi-focus	LP-PCNN [5]	7.060268	0.016244	42.227196	1.6965
	NSCT-PCNN [23]	7.040197	0.015665	40.550226	216.167
	Ours	7.104631	0.016058	41.508493	6.754
CT/MRI	LP-PCNN [5]	5.690078	0.082494	63.087874	0.331
	NSCT-PCNN [23]	6.838788	0.077755	58.903765	46.815
	Ours	6.969029	0.080016	60.576231	1.425
multi-spectral and full-color	LP-PCNN [5]	6.479447	0.209883	35.579854	34.625
	NSCT-PCNN [23]	7.227013	0.232942	46.672342	4710.249
	Ours	7.227010	0.232942	46.672549	116.220

7. Conclusion

In this study, we propose a Riesz transform based approach to image fusion. Specifically, an image is decomposed by Riesz Laplacian spline wavelet pyramid to derive a new representation, with which PCNN optimization is employed to obtain fused images. Through subjective observation and objective index analysis, we observe that our method demonstrates superior performance in image fusion with low time complexity. Through experiments, we find that our method can be further optimized by selecting PCNN parameters adaptively. Another future work is to choose Riesz transform as the objective function of fusion rules to improve fusion effects further.

Acknowledgement

This work is supported by the National Natural Science Foundation of China (NSFC) under contract No. 61871258.

References

[1] European Space Agency and H. Kramer. Satellite Missions catalogue. `https://directory.eoportal.org/web/eoportal/satellite-missions`.

[2] S. Bhat and D. Koundal. Multi-focus image fusion using neutrosophic based wavelet transform. *Applied Soft Computing*, 106:107307, 2021. doi:10.1016/j.asoc.2021.107307.

[3] T. A. Bui and X. T. Duong. Higher-order Riesz transforms of Hermite operators on new Besov and Triebel–Lizorkin spaces. *Constructive Approximation*, 53:85–120, 2021. doi:10.1007/s00365-019-09493-y.

[4] J. Chen, L. Chen, and M. Shabaz. Image fusion algorithm at pixel level based on edge detection. *Journal of Healthcare Engineering*, page 5760660, 2021. doi:10.1155/2021/5760660.

[5] K. J. He, X. Jin, R. Nie, D. M. Zhou, Wang Q., and Yu J. Color image fusion based on simplified PCNN and Laplace pyramid decomposition (in Chinese). *Journal of Computer Applications*, 2016(S1):133–137, 2016.

[6] X. Jin, R. Nie, and D. M. Zhou. Color image fusion researching based on S-PCNN and Laplacinan pyramid. In *Proc. 2nd Int. Conf. Cloud Computing and Big Data in Asia CloudCom-Asia 2015*, volume 9106 of *Lecture Notes in Computer Science*, pages 179–188, Huangshan, China, 17-19 Jun 2015. doi:10.1007/978-3-319-28430-9_14.

[7] K. A. Johnson and J. A. Becker. The whole brain Atlas. `https://www.med.harvard.edu/AANLIB/`. Harvard Medical School.

[8] W. Li, Q. Liu, K. Wang, and K. Cai. Improving medical image fusion method using fuzzy entropy and nonsubsampling contourlet transform. *International Journal of Imaging Systems and Technology*, 31(1):204–214, 2021. doi:10.1002/ima.22476.

[9] Y. Liu, L. Wang, J. Cheng, C. Li, and X. Chen. Multi-focus image fusion: A survey of the state of the art. *Information Fusion*, 64:71–91, 2020. doi:10.1016/j.inffus.2020.06.013.

[10] Y. F. Lu and T. Zhang. Image quality assessment method via Riesz-transform based structural similarity. *Chinese Journal of Liquid Crystals and Displays*, 30(6):992–999, 2015. doi:10.3788/YJYXS20153006.0992.

[11] S. Moritoh and N. Takemoto. Expressing Hilbert and Riesz transforms in terms of wavelet transforms. *Integral Transforms and Special Functions*, 34(5):365–370, 2023. doi:10.1080/10652469.2022.2126465.

[12] C. Panigrahy, A. Seal, and N. K. Mahato. MRI and SPECT image fusion using a weighted parameter adaptive dual channel PCNN. *IEEE Signal Processing Letters*, 27:690–694, 2020. doi:10.1109/LSP.2020.2989054.

[13] C. Panigrahy, A. Seal, and N. K. Mahato. Parameter adaptive unit-linking dual-channel PCNN based infrared and visible image fusion. *Neurocomputing*, 514:21–38, 2022. doi:10.1016/j.neucom.2022.09.157.

[14] A. I. Rahmani, M. Almasi, N. Saleh, and M. Katouli. Image fusion of noisy images based on simultaneous empirical wavelet transform. *Traitement du Signal*, 37(5):703–710, 2020. doi:10.18280/ts.370502.

[15] S. Savić. Multifocus Image Fusion. `https://dsp.etfbl.net/mif/`. Chair of General Electrical Engineering, Faculty of Electrical Engineering, University of Banja Luka, Republic of Srpska.

[16] S. Savić and Z. Babić. Multifocus image fusion based on the first level of empirical mode decomposition. In *Proc. 19th Int. Conf. Systems, Signals and Image Processing IWSSIP 2012*, pages 604–607, Vienna, Austria, 11-13 Apr 2012. IEEE. `https://ieeexplore.ieee.org/abstract/document/6208315`.

[17] S. Singh, N. Mittal, and H. Singh. A feature level image fusion for IR and visible image using mNMRA based segmentation. *Neural Computing and Applications*, 34(10):8137–8145, 2022. doi:10.1007/s00521-022-06900-7.

[18] J. Sun, Q. Han, K. Liang, L. Zhang, and Z. Jin. Multi-focus image fusion algorithm based on Laplacian pyramids. *Journal of the Optical Society of America A*, 35(3):480–490, 2018. doi:10.1364/JOSAA.35.000480.

[19] L. F. Tang, H. Zhang, H. Xu, and J. Y. Ma. Deep learning-based image fusion: A survey. *Journal of Image and Graphics*, 28(1):3–36, 2023. doi:10.11834/jig.220422.

[20] The MathWorks, Inc. MATLAB. Natick, MA, USA. [Accessed: 2018]. `https://www.mathworks.com`.

[21] X. Tian, W. Zhang, Y. Chen, Z. Wang, and J. Ma. Hyperfusion: A computational approach for hyperspectral, multispectral, and panchromatic image fusion. *IEEE Transactions on Geoscience and Remote Sensing*, 60:5518216, 2021. doi:10.1109/TGRS.2021.3128279.

[22] Y. Xing, S. Yang, Z. Feng, and L. Jiao. Dual-collaborative fusion model for multispectral and panchromatic image fusion. *IEEE Transactions on Geoscience and Remote Sensing*, 60:5400215, 2020. doi:10.1109/TGRS.2020.3036625.

[23] J. J. Yang. Improved gradient image fusion algorithm based on NSCT and PCNN. *Digital Technology and Application*, (11):124–127, 2015. doi:10.19695/j.cnki.cn12-1369.2015.11.092.

Lung and Colon Cancer Detection from CT Images Using Deep Learning

Joseph D. Akinyemi[1], Akinkunle A. Akinola[2], Olajumoke O. Adekunle[1], Taiwo O. Adetiloye[3], Emmanuel J. Dansu[4,5]

[1]*Department of Computer Science, University of Ibadan, Ibadan, Nigeria*

[2]*Department of Data Science, Bowling Green State University, Bowling Green, Ohio, USA*

[3]*BigCodeGen LLC, Fort Worth, Texas, USA*

[4]*Department of Mathematical Sciences, Federal University of Technology, Akure, Nigeria*

[5]*Ecological Integration Laboratory, Graduate School of Life Sciences, Tohoku University, Sendai, Japan*

jd.akinyemi@ui.edu.ng

Abstract. Cancer is a deadly disease that has gained a reputation as a global health concern. Further, lung cancer has been widely reported as the most deadly cancer type globally, while colon cancer comes second. Meanwhile, early detection is one of the primary ways to prevent lung and colon cancer fatalities. To aid the early detection of lung and colon cancer, we propose a computer-aided diagnostic approach that employs a Deep Learning (DL) architecture to enhance the detection of these cancer types from Computed Tomography (CT) images of suspected body parts. Our experimental dataset (LC25000) contains 25 000 CT images of benign and malignant lung and colon cancer tissues. We used weights from a pre-trained DL architecture for computer vision, EfficientNet, to build and train a lung and colon cancer detection model. EfficientNet is a Convolutional Neural Network architecture that scales all input dimensions such as depth, width, and resolution at the same time. Our research findings showed detection accuracies of 99.63%, 99.50%, and 99.72% for training, validation, and test sets, respectively.

Key words: cancer detection, efficientNet, CT images, healthcare.

1. Introduction

Cancer is one of the leading causes of death globally causing about 10 million deaths or one out of every six deaths in the year 2020. Of the various types of cancer, lung cancer accounts for the highest number of deaths, accounting for about 18% of cancer deaths in the year 2020; the second highest being colon and rectum cancer, which account for 9% of cancer deaths [4, 22]. The presence of cancer in the body is often medically confirmed by medical imaging techniques such as Computed Tomography (CT) scan or Magnetic Resonance Imaging (MRI) which are then analyzed and interpreted by medical experts. To improve detection accuracy and reduce doctors' burdens, Machine Learning (ML) and Deep Learning (DL) techniques have been employed to accelerate the detection of cancer in such medical images. This enables researchers to analyze a large number of patients in much less time and at a lower cost [18]. The accuracy of these models on real-world data has been limited largely because of insufficient datasets available for experimentation, yet, near-perfect accuracy is highly desirable in medical systems.

DL models, especially for computer vision, thrive on large volumes of data. When these are not available, model performance may be unsatisfactory. Until 2019, freely available cancer image datasets have only contained a few thousand images. However, since the release of the LC25000 dataset [3], the research community has seen consistent progress in the accuracy of cancer detection and classification models. However, our major observation is that most published works have performed experiments on subsets of the dataset (either lung cancer or colon cancer). This may be because of the relatively large size of the entire dataset which makes attempts to build a single model to detect both lung and colon cancers not achieve near-perfect accuracy. In this work, a single DL model was developed to detect and classify lung and colon cancer with improved classification accuracy over existing models. We used the pre-trained EfficientNet [19] model to build a lung and colon cancer detection/classification model and obtained 99.72% accuracy on a hold-out (out-of-training) test set of 5000 images in only 44 training epochs. The introspection results of our training and validation also reveal a good training progression and "goodness of fit". Our motivations for developing a single model for lung and colon cancer classification include (i) the fact that both diseases have the highest cancer fatalities, and (ii) the possibility of achieving better accuracy in medical predictions when large datasets are available.

Our research contribution is to develop a novel DL model based on EfficientNet for accurate and robust lung and colon cancer detection. This model is designed to automatically extract the most important features from medical images for accurate and reliable cancer detection. Furthermore, the proposed model is capable of achieving high accuracy and robustness in the detection of lung and colon cancer. Additionally, we have evaluated the proposed model on the LC25000 dataset to demonstrate its effectiveness in detecting both types of cancer.

The rest of this paper is organized as follows. In Section 2, we review literature related to the application of DL in cancer detection. The materials and methods used in the research are given in Section 3. This includes the dataset used, the network architecture, and applicable experiments. Section 4 presents our results and discussions. Finally, conclusions and future work are given in Section 5.

2. Review of related literature

The growing success of ML and DL has made it applicable in several walks of life including cancer detection. In this section, we present a review of recent relevant works in lung and colon cancer detection.

A ML-based lung and colon cancer detection using hybrid model comprising deep feature extraction and ensemble learning was introduced by [18]. Their hybrid model which evaluated the histopathological (LC25000) lung and colon datasets achieved accuracy rates of 99.05%, 100%, and 99.30%, respectively. This feature extraction approach

(a hybrid ensemble feature extraction model that combines deep feature extraction and high-performance ensemble learning for cancer images) is anticipated to be helpful for the diagnosis of lung and colon cancers in medical sectors.

In [8], eight different pre-trained models for lung and colon cancer classification were employed, also using the LC25000 dataset. Five of the eight pre-trained models (Inception V2 & V3, MobileNet, Xception and DenseNet169) achieved 100% accuracy, while the rest (VGG-16, ResNet50 and NASNetMobile) achieved more than 95% accuracy. While this may be a pointer to the effectiveness of these pre-trained models in classifying lung and colon cancer images, it must be stated that the reported results are for binary classifications on subclasses of the dataset with the largest subclass containing 10 000 images. [14] proposed research directions to assist in early-stage detection of cancer while identifying gaps for future development of lung cancer detection in medical IoT devices based on various ML algorithms utilized for detecting a number of diseases.

In [15], a Convolutional Neural Network (CNN) model was proposed that was characterized by its speed of diagnosis and high accuracy with few parameters for diagnosing colon cancer. The model consists of two paths where each path is responsible for creating 256 feature maps to increase the number of features at different levels in order to improve the accuracy and sensitivity of the classification. They compared the performance of their CNN model with VGG-16 model on the 10 000 colon CT images of the LC25000 dataset and reported classification accuracies of 99.6% and 96.2%, respectively.

Having developed a lung cancer detection model using InceptionV3, Histogram of Gradients (HoG) and Daisy feature extraction, [6] obtained 99.6% accuracy in classifying benign and malignant lung tissues using 15000 lung CT scans from the LC25000 dataset. [12] introduced a lung cancer classification using Particle Swarm Optimization (PSO), Genetic Algorithm (GA), and Support Vector Machine (SVM). In terms of classification, PSO-GA-SVM outperformed SVM without parameter optimization. The accuracy, precision, recall, and F1-score values for the PSO-GA-SVM, were discovered to be 97.69%, 98.46%, 98.82%, and 97.66%, respectively.

A DL-based classification framework for lung and colon cancer diagnosis using ML was presented in [11]. A framework was proposed that can help medical professionals identify as well as differentiate among five types of lung and colon tissues. A supervised learning approach with a DL model was used to identify three cancerous and two non-cancerous lung and colon tumors. In that research, pathological images were obtained, relating to these types of cancers from the LC25000 dataset, which was also used to train and validate the approach. After obtaining approximately 96% accuracy and proving the superiority of this method over other similar cancer detection methods, it was concluded that this computer-based identification method would allow less costly pathologists' diagnoses of lung and colon cancer cases with minimal effort and time.

In [20], three CNN models were trained and tested for colon cancer detection using the 10 000 colon CT images of the LC25000 dataset. Two of the three CNN models

were built from scratch while one of them was a pre-trained CNN (MobileNetV2). Of the two models built from scratch, one used average-pooling and reported an accuracy of 95.48%, while the other used max-pooling and reported an accuracy of 97.49%, but the pre-trained MobileNetV2 outperformed them both with a classification accuracy of 99.67%. All accuracies were reported on 20% of the dataset while the remaining 80% were used to train the models.

In [2], a framework was proposed based on multiple lightweight DL models (ShuffleNet, MobileNet and SqueezeNet) for the detection of lung and colon cancer from CT images in the LC25000 dataset. Following the extraction of deep features by these lightweight DL models, features transformations such as Principal Component Analysis (PCA) and Fast Walsh-Hadamard Transform were employed to reduce the dimension of the features and extract a relevant subset of dense features. The resulting feature subsets from the two transforms were then used to train four ML models out of which Support Vector Machine had the best accuracy of 99.6%.

In [9], seven different DL architectures were tested on the colon images subset of the LC25000 dataset. One of the DL architectures was a 9-layer CNN consisting of convolution layers, max-pooling layers, a flatten layer, a dropout layer, and dense layers. The other six models were pre-trained models (VGG-16, EfficientNetB0, ResNet101V2, ResNet50, DenseNet121, and MobileNetV2). All the models were trained with 80% of the data, and validated with 10% of the data while the remaining 10% was held out for testing. The authors reported that their 9-layers CNN had the best accuracy of 99.8% and thus outperformed all the pre-trained models. While this is commendable, it is not justifiable from a computational standpoint, especially because they did not provide details of how the pre-trained models were used for transfer learning (hyperparameter tuning, whether layers were added, retrained etc.). However, the performance of this 9-layer CNN is commendable, the training and validation plots show good generalization, yet we believe that the accuracy of this 9-layer architecture is likely to drop when used for a multi-class classification like ours.

To develop a transfer-learning model for detecting lung and colon cancer using LC25000 histopathological images, [13] tuned and used a pre-trained model, AlexNet, to classify the CT images of lung and colon tissues. Initially, the model achieved an overall accuracy of 89.9%. A so-called class-selective image processing method was employed to identify the underperforming class and selectively preprocess the images in that class. This method improved the model accuracy to 98.4% showing that simple and efficient image processing methods can improve ML model performance.

In [17], an analysis of lung cancer using a deep neural network was presented. 15 000 samples of histopathological photographs of lung adenocarcinoma, lung squamous cell carcinoma, and benign lung tissue from three different types were used. Histopathological photographs of lung tissues were classified using a Computer-Aided Diagnostic (CAD)

method. Using various pre-trained models with hyper-tuning, the best accuracy was achieved from ResNet 101, a CNN network, at 98.67%.

In [23], a method for detecting and classifying colon cancer was proposed, using what was called MA_ColonNET. This MA_ColonNET was built from a 45-layer CNN architecture consisting of 2D convolution layers and max-pooling layers. MA_ColonNET achieved a 99.75% accuracy on 2000 lung CT images which were used for testing the trained model.

From the reviewed works, it has been observed that achieving very high accuracies (over 99%) on "out-of-training" samples has been relatively difficult when building a single model to predict both lung and colon cancer, compared with separate models for lung and colon cancer detection. This is most likely due to the differences in the image structures between the different disease types. However, considering the computational demand of training, validating, testing, and deploying DL models, it is more expensive to train two separate models on one dataset than it is to train a single model. More so, since the LC25000 dataset has come as an answer to a request for larger datasets for medical image analysis, it is sensible for the computer vision research community to take advantage of its large size to develop better predictive models.

3. Materials and methods

In this work a DL approach for cancer detection from CT images is proposed. We have experimented with a relatively large dataset of 25 000 images, details of which can be found in section 3.1. Our DL architecture employs the pre-trained EfficientNet-B7 [19] as the backbone and adds a few dense layers on top of it. The details of the architecture are presented in section 3.2. The experimental setup is presented in section 3.3.

3.1. Dataset

The dataset used in this work is the lung and colon cancer histopathological dataset [3] which is also referred to as LC25000. It originally contains 250 images each of *Lung benign tissue* (`Lung_n`), *Lung adenocarcinoma* (`Lung_aca`), *Lung squamous cell carcinoma* (`Lung_scc`), *Colon adenocarcinoma* (`Colon_aca`), and *Colon benign* (`Colon_n`) tissue totaling 750 images which were then augmented to 25 000 images with a total of 5000 images in each of the five classes. For the purpose of testing the performance of our network, we created a hold-out test set that contained 20% of the total size of the dataset (i.e. 5000 images). The remaining 20 000 images were used for training and validation in order to experiment and improve the architecture before finally testing it on the test set. The validation split is 10% of the remaining 20 000 images (i.e. 2000 images), while the remaining 18 000 images were used for training.

It is noteworthy that the hold-out test set was created with an equal number of

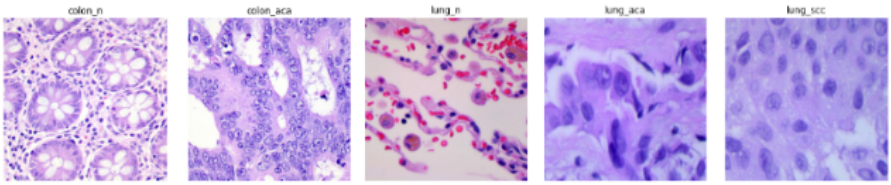


Fig. 1. Samples of CT images from the LC25000 dataset [3].

Layer (type)	Output Shape	Param #	Connected to
global_average_pooling2d_2 (Glo	(None, 2560)	0	top_activation[0][0]
flatten_2 (Flatten)	(None, 2560)	0	global_average_pooling2d_2[0][0]
dense_8 (Dense)	(None, 256)	655616	flatten_2[0][0]
dense_9 (Dense)	(None, 128)	32896	dense_8[0][0]
dropout_2 (Dropout)	(None, 128)	0	dense_9[0][0]
dense_10 (Dense)	(None, 64)	8256	dropout_2[0][0]
dense_11 (Dense)	(None, 5)	325	dense_10[0][0]

Total params: 64,794,780
Trainable params: 697,093
Non-trainable params: 64,097,687

Fig. 2. Architecture of the top layers attached to EfficientNet-B7 for cancer detection from CT images.

images (1000) from each of the five classes. In the same vein, the validation split was done so that equal quantities (400) of images from each class are present in the split. This balance is important to ensure that class imbalance does not bias the classification probabilities of each class at training. Figure 1 shows samples of images in the dataset – one from each of the five classes.

3.2. Network Architecture

The EfficientNet-B7 architecture was developed by the Google Research Brain team. It was first presented in the paper by [19] in which a family of neural network models (B0 to B7) was developed by uniformly scaling all convolution dimensions of depth, width, and resolution using compound scaling with a fixed ratio.

While EfficientNet-B7 showed improved accuracy on the ImageNet dataset [7] at that time (84.4% accuracy), the architecture also had fewer mode parameters compared to the state-of-the-art. This means that it can be trained in less time and with fewer computing resources and these are desirable features for transfer learning.

To create our cancer detection architecture, we reused the pre-trained EfficientNet-B7 without its top layer. Then, we added 4 dense layers on top of the pre-trained architecture as shown in Figure 2. The EfficientNet model architecture is very deep due

Tab. 1. Hyperparameter settings.

Hyperparameter	Value
Input image size	$224 \times 224 \times 3$
Base learning rate	$5 \cdot 10^{-5}$
Batch size	128
Optimizer	Adam
Seed	42
Loss function	Categorical cross entropy
Evaluation metric	Accuracy

Tab. 2. Accuracy and loss values.

	Training	Validation	Testing
Accuracy [%]	99.63	99.50	99.72
Loss	0.0124	0.0145	0.0103

to the compound scaling method adopted. The architecture was trained with images of dimension $224 \times 224 \times 3$, thus, that is the input dimension specified in the input layer of the architecture.

The final dense layer of the network has five units for classifying the input image into one of the five classes. The network was trained with a base learning rate of $5 \cdot 10^{-5}$, using the Adam optimizer and a categorical cross-entropy loss function. We also included a dropout layer to drop 30% of the neurons in the previous layer in order to combat overfitting and obtain a robust model. The model was trained with early stopping for 44 epochs. We set a seed value of 42 to ensure reproducibility in the random selection of the test set. The modified EfficientNet-B7 architecture was trained on 18 000 CT images, validated on 2000 CT images and tested on a hold-out set of 5000 images.

3.3. Experimental setup

Experiments were carried out on Kaggle's GPU $T4 \times 2$ and the Tensorflow library was employed for DL. Table 1 shows the hyperparameter settings employed for the experiments.

4. Results and discussion

The results of training, validation, and testing are presented in Table 2. From the table, it can be observed that the model training and prediction are in step and it is not overfitting.

Tab. 3. Comparison of cancer prediction models on LC25000 dataset.

No.	Ref.	Cancer type	ML/DL model	Validation protocol	Acc. [%]
1	[8]	Lung and colon	Inception, Xception, VGG-16, ResNet, MobileNet, DenseNet, NASNetMobile	Train (80%); val. (20%)	96-100
2	[15]	Colon	VGG-16	Train (70%); val. (30%)	99.6
3	[20]	Colon	MobileNetV2	Train (80%); val. (20%)	99.67
4	[17]	Lung	ResNet 101	Train (80%); val. (20%)	98.67
5	[23]	Colon	MA_Colon NET	Train (80%); val. (20%)	99.75
6	[21]	Lung and colon	DarkNet+ SVM	Train (70%); val. (30%)	99.69
7	[10]	Lung	CNN	Train (90%); val. (10%)	97.2
8	[16]	Colon	DenseNet, ResNet, SVM, RF, KNN, XGB	Train (75%); val. (25%)	98.53
9	[5]	Lung	Ensemble	Train (80%); val. (20%)	99.6
10	[1]	Lung and colon	DHS-CapsNet	Train (70%); val. (15%); test (15%)	99.23
11	Ours	Lung and colon	EfficientNetB7	Train (∼70%); val. (∼10%); test (20%)	99.72

The accuracy and loss plots in Figures 3 and 4 are a confirmation of the fact that the model is not overfitting. They show a smooth progression of training towards convergence and this is an indication of the robustness of the EfficientNet architecture for transfer learning on a task such as cancer detection from CT images.

Figure 5 shows the precision, recall, and F1-score values on the test set, and Figures 6 and 7 show the confusion matrix with and without normalization of the predictions on the test set. These two figures again show the details of the predictive power of the developed cancer detection model as there were very negligible misclassifications at test time. It is noteworthy that this model performance was achieved without further data augmentation on the 18000 training images and in 44 training epochs. This is an indication that the compound scaling method of EfficientNet is effective in detecting cancerous cell nodules in tissue CT images.

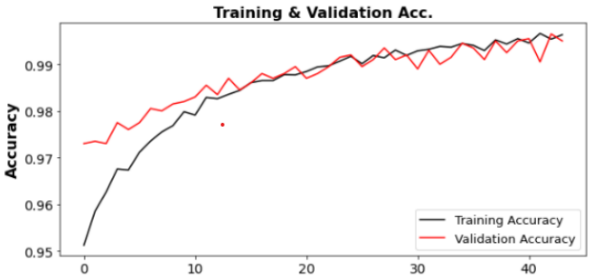


Fig. 3. Training and validation accuracy.

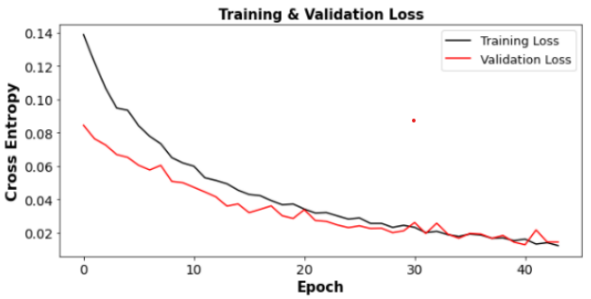


Fig. 4. Training and validation loss.

	precision	recall	f1-score	support
0	0.9990	0.9970	0.9980	1000
1	1.0000	1.0000	1.0000	1000
2	0.9930	0.9950	0.9940	1000
3	0.9990	0.9980	0.9985	1000
4	0.9950	0.9960	0.9955	1000
accuracy			0.9972	5000
macro avg	0.9972	0.9972	0.9972	5000
weighted avg	0.9972	0.9972	0.9972	5000

Fig. 5. Precision, recall and F1-score values.

We compared the performance of our developed model with other existing models on the LC25000 dataset. Table 3 shows our comparative analysis of works published between the years 2020 and 2022 on cancer detection from CT images using the LC25000 dataset. Most of the works have employed various pre-trained DL models, but none had employed the EfficientNet model for this task. Also, it can be observed that our developed model

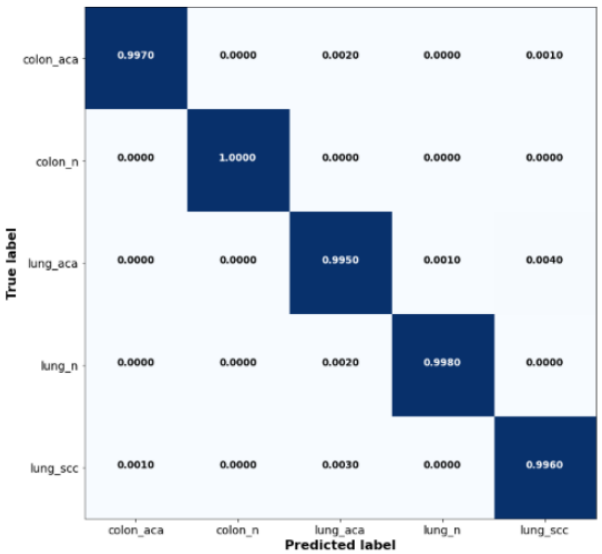


Fig. 6. Normalized confusion matrix.

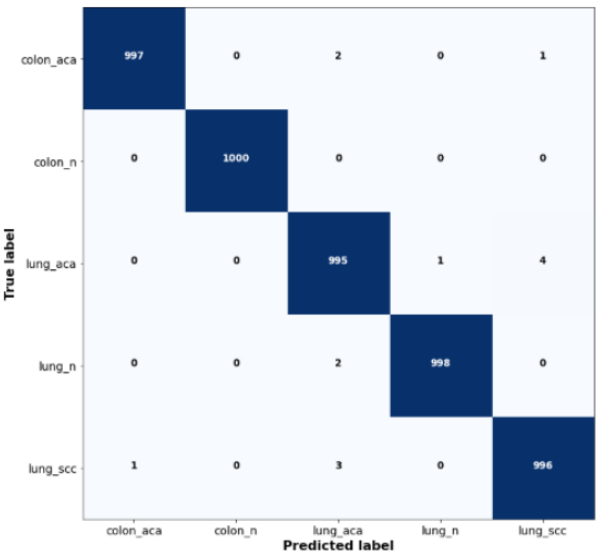


Fig. 7. Confusion matrix without normalization.

significantly outperforms a good number of the works presented. However, a few works, which performed at par with our work, have been closely 'x-rayed' and our analysis and discoveries are presented in the next paragraph.

The paper [8] reported a 100% accuracy on the LC25000 dataset, but upon careful examination, we found that they performed three binary classification tasks on three subsets of the dataset and each of these subsets contains 10000 images. Therefore, their classification model cannot be said to be as robust as ours. [15] used only the colon cancer subset of the LC25000 dataset which contained only 10000 images. [21] used only training and validation split and this method has been shown to result in overfitting to the validation set. More so, the DarkNet part of their model was trained for 3000 epochs and then some feature optimizations were carried out on the features before they were used by SVM for a final classification. This does not compare to our model which was trained for just 44 epochs yet achieved higher accuracy. As for [23], their work was also built on the 10 000 images of colon tissues, which is just a binary classification task and does not come near our multi-class classification of 5 classes ranging from lung to colon cancer.

5. Conclusions and future work

While the research in the detection of lung cancer using ML and DL seems to be getting more attention than colon cancer, both deadly diseases ultimately require early detection and diagnosis by health practitioners in order to mitigate the spread to other parts of the body. The use of ML and DL methods to aid in the diagnosis of these diseases will not only reduce the burdens of relevant stakeholders in cancer diagnosis, but it also holds great prospects for faster and more accurate diagnosis, thus resulting in fewer fatalities.

In this work, a robust DL method for lung and colon cancer has been developed. Trained on only 18 000 images, the model was able to detect cancer in a hold-out test set containing 5000 images to a very high degree of accuracy (99.72%). The fact that existing works also have similar accuracies on the same dataset is a pointer to the fact that we are drawing closer to a breakthrough in AI-assisted diagnosis of cancer disease.

Future work should focus on experimenting on a larger corpus, preferably from heterogeneous sources (e.g., various hospitals, various machines, etc.). More work also needs to be done in terms of interpretation and analysis of the predictions to medical experts and patients in order to improve the trust and confidence in AI-assisted diagnosis of cancer.

Acknowledgments

We would like to express our gratitude to BigCodeGen LLC for its generous support of our research project. We also appreciate the reviewers for their valuable comments.

References

[1] K. Adu, Y. Yu, J. Cai, K. Owusu-Agyemang, B. A. Twumasi, and X. Wang. DHS-CapsNet: Dual horizontal squash capsule networks for lung and colon cancer classification from whole slide histopathological images. *International Journal of Imaging Systems and Technology*, 31(4):2075–2092, 2021. doi:10.1002/ima.22569.

[2] O. Attallah, M. F. Aslan, and K. Sabanci. A framework for lung and colon cancer diagnosis via lightweight deep learning models and transformation methods. *Diagnostics*, 12(12), 2022. doi:10.3390/diagnostics12122926.

[3] A. A. Borkowski, M. M. Bui, L. B. Thomas, C. P. Wilson, L. A. DeLand, and S. M. Mastorides. LC25000 Lung and colon histopathological image dataset. 2019. `https://github.com/tampapath/lung_colon_image_set`.

[4] CDC. An update on cancer deaths in the united states. In *Center for Disease Control (CDC)*. 2022. `https://www.cdc.gov/cancer/dcpc/research/update-on-cancer-deaths/`.

[5] H. Chen, X. Xu, T. Ge, C. Hua, X. Zhu, Q. Wang, Z. Yu, and R. Zhang. A novel tool for the risk assessment and personalized chemo-immunotherapy response prediction of adenocarcinoma and squamous cell carcinoma lung cancer. *International Journal of General Medicine*, 14:5771–5785, 2021. doi:10.2147/IJGM.S327641.

[6] M. Chen, S. Huang, Z. Huang, and Z. Zhang. Detection of lung cancer from pathological images using CNN model. In *Proc. 2021 IEEE Int. Conf. Computer Science, Electronic Information Engineering and Intelligent Control Technology (CEI)*, pages 352–358, Fuzhou, China, 24-26 Sep 2021. doi:10.1109/CEI52496.2021.9574590.

[7] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A large-scale hierarchical image database. In *Proc. 2009 IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, pages 248–255, Miami, FL, USA, 20-25 Jun 2009. doi:10.1109/CVPRW.2009.5206848.

[8] S. Garg and S. Garg. Prediction of lung and colon cancer through analysis of histopathological images by utilizing pre-trained CNN models with visualization of class activation and saliency maps. In *Proc. 3rd Artificial Intelligence and Cloud Computing Conf. (AICCC) 2020*, ACM International Conference Proceeding Series, pages 38–45. Association for Computing Machinery, Kyoto, Japan, Dec 18-20 2020. doi:10.1145/3442536.3442543.

[9] Md I. Hasan, Md S. Ali, Md H. Rahman, and Md K. Islam. Automated detection and characterization of colon cancer with deep convolutional neural networks. *Journal of Healthcare Engineering*, 2022:5269913, 2022. doi:10.1155/2022/5269913.

[10] B. K. Hatuwal and H. C. Thapa. Lung cancer detection using convolutional neural network on histopathological images. *International Journal of Computer Trends and Technology*, 68(10):21–24, 2020. doi:10.14445/22312803/IJCTT-V68I10P104.

[11] M. Masud, N. Sikder, A. A. Nahid, A. K. Bairagi, and M. A. Alzain. A machine learning approach to diagnosing lung and colon cancer using a deep learning-based classification framework. *Sensors*, 21(3):748, 2021. doi:10.3390/s21030748.

[12] F. Maulidina, Z. Rustam, and J. Pandelaki. Lung Cancer Classification using Support Vector Machine and Hybrid Particle Swarm Optimization-Genetic Algorithm. In *Proc. 2021 Int. Conf. Decision Aid Sciences and Application (DASA)*, pages 751–755, Sakheer, Bahrain, 07-08 Dec 2021. doi:10.1109/DASA53625.2021.9682259.

[13] S. Mehmood, T. M. Ghazal, M. A. Khan, M. Zubair, M. T. Naseem, T. Faiz, and M. Ahmad. Malignancy detection in lung and colon histopathology images using transfer learning with class selective image processing. *IEEE Access*, 10:25657–25668, 2022. doi:10.1109/ACCESS.2022.3150924.

[14] K. Pradhan and P. Chawla. Medical Internet of things using machine learning algorithms for lung cancer detection. *Journal of Management Analytics*, 7(4):591–623, 2020. doi:10.1080/23270012.2020.1811789.

[15] Y. Qasim, H. Al-Sameai, O. Ali, and A. Hassan. Convolutional neural networks for automatic detection of colon adenocarcinoma based on histopathological images. In F. Saeed, F. Mohammed, and A. Al-Nahari, editors, *Innovative Systems for Intelligent Health Informatics: Proc. Int. Conf. Reliable Information and Communication Technology (IRICT)*, volume 72 of *Lecture Notes on Data Engineering and Communications Technologies*, pages 19–28. Springer, Cham, 22-23 Dec 2021. doi:10.1007/978-3-030-70713-2_3.

[16] D. Sarwinda, A. Bustamam, R. H. Paradisa, T. Argyadiva, and W. Mangunwardoyo. Analysis of deep feature extraction for colorectal cancer detection. In *Proc. 2020 4th Int. Conf. Informatics and Computational Sciences (ICICoS)*, pages 1–5, Semarang, Indonesia, 10-11 Nov 2020. doi:10.1109/ICICoS51170.2020.9298990.

[17] S. Shandilya and S. R. Nayak. Analysis of lung cancer by using deep neural network. In M. Mishra, R. Sharma, Rathore A. K., J. Nayak, and B. Naik, editors, *Proc. 2nd Conf. Innovation in Electrical Power Engineering, Communication, and Computing Technology (IEPCCT 2021)*, volume 814 of *Lecture Notes in Electrical Engineering*, pages 427–436, Bhubaneswar, India, 24–26 Sep 2022. Springer, Singapore. doi:10.1007/978-981-16-7076-3_37.

[18] Md A. Talukder, Md M. Islam, Md A. Uddin, A. Akhter, K. F. Hasan, and M. A. Moni. Machine learning-based lung and colon cancer detection using deep feature extraction and ensemble learning. *Expert Systems with Applications*, 205:117695, 2022. doi:10.1016/j.eswa.2022.117695.

[19] Z. Tan, Y. Yang, J. Wan, G. Guo, and S. Z. Li. Deeply-learned hybrid representations for facial age estimation. In *Proc. 28th Int. Joint Conf. Artificial Intelligence (IJCAI)*, pages 3548–3554, Macao, China, 10-16 Aug 2019. doi:10.24963/ijcai.2019/492.

[20] Z. Tasnim, S. Chakraborty, F. M. J. M. Shamrat, A. N. Chowdhury, H. A. Nuha, A. Karim, S. B. Zahir, and Md M. Billah. Deep learning predictive model for colon cancer patient using CNN-based classification. *International Journal of Advanced Computer Science and Applications*, 12(8):687–696, 2021. doi:10.14569/IJACSA.2021.0120880.

[21] M. Toğaçar. Disease type detection in lung and colon cancer images using the complement approach of inefficient sets. *Computers in Biology and Medicine*, 137:104827, 2021. doi:10.1016/j.compbiomed.2021.104827.

[22] WHO. Cancer: Key facts, 7 Feb 2022. `https://www.who.int/news-room/fact-sheets/detail/cancer`.

[23] M. Yildirim and A. Cinar. Classification with respect to colon adenocarcinoma and colon benign tissue of colon histopathological images with a new CNN model: MA_ColonNET. *International Journal of Imaging Systems and Technology*, 32(1):155–162, 2022. doi:10.1002/ima.22623.

Performance Evaluation of Machine Learning Models to Predict Heart Attack

Majid Khan[1], Ghassan Husnain[1*], Waqas Ahmad[1], Zain Shaukat[1], Latif Jan[1], Ihtisham Ul Haq[2], Shahab Ul Islam[1], Atif Ishtiaq[1]

[1] *Department of Computer Science, Iqra National University Peshawar, 25100, Pakistan*

[2] *Department of Mechatronics Engineering, University of Engineering and Technology, Peshawar, 25100, Pakistan*

[*] *Corresponding author: Ghassan Husnain (email: ghassan.husnain@gmail.com)*

Abstract. Coronary Artery Disease is the type of cardiovascular disease (CVD) that happens when the blood vessels which stream the blood toward the heart, either become tapered or blocked. Of this, the heart is incapable to push sufficient blood to encounter its requirements. This would lead to angina (chest pain). CVDs are the leading cause of mortality worldwide. According to WHO, in the year 2019 17.9 million people deceased from CVD. Machine Learning is a type of artificial intelligence that uses algorithms to help analyse large datasets more efficiently. It can be used in medical research to help process large amounts of data quickly, such as patient records or medical images. By using Machine Learning techniques and methods, scientists can automate the analysis of complex and large datasets to gain deeper insights into the data. Machine Learning is a type of technology that helps with gathering data and understanding patterns. Recently, researchers in the healthcare industry have been using Machine Learning techniques to assist with diagnosing heart-related diseases. This means that the professionals involved in the diagnosis process can use Machine Learning to help them figure out what is wrong with a patient and provide appropriate treatment. This paper evaluates different machine learning models performances. The Supervised Learning algorithms are used commonly in Machine Learning which means that the training is done using labelled data, belonging to a particular classification. Such classification methods like Random Forest, Decision Tree, K-Nearest Neighbour, XGBoost algorithm, Naïve Bayes, and Support Vector Machine will be used to assess the cardiovascular disease by Machine Learning.

Key words: cardiovascular disease, Machine Learning, heart attack, prediction.

1. Introduction

The heart is a very important organ of our body and most diseases involve the heart in some way. Heart disease refers to any condition that affects the heart and its ability to pump oxygen-rich blood to other parts of the body. There are different types of heart diseases, including coronary artery disease, congenital heart disease, and arrhythmias. Common symptoms can include chest pain, dizziness, and sweating. Risk factors for heart disease include smoking, high blood pressure, diabetes, and obesity [20]. Early monitoring and detection of heart diseases can greatly reduce the mortality rate. However, many people seek medical attention from a heart specialist only after the disease has progressed significantly [15]. Research on new therapeutic drug agents is ongoing (see for example [7]) but predicting a dangerous situation before it occurs is highly desirable.

Predictions about whether someone is likely to develop a heart disease in the future are therefore incredibly important. Unfortunately, these predictions are often not accurate, leading to premature death. To help with this problem, Artificial Intelligence is used to create algorithms that can process data in a way that is similar to processing performed by humans. Biological factors as training data for machine learning algorithms are used in order to teach a machine to recognize patterns that occur in nature. Examples of these biological factors include chest pain, angina, hypertension, age, cholesterol, blood pressure, sex, etc. By using these data, the machine learning algorithms are provided with enough information to accurately predict patterns in the real world [13]. In this paper, we are using various factors from biology, like cholesterol, blood pressure, sex, and age, as the data used in the experiments. The aim is to use these data to train machine learning algorithms, such as decision trees, linear regression, K-Nearest Neighbour classifiers (K-NN), and Support Vector Machine (SVM). We are trying to prove which algorithm can offer the best accuracy with these data sets. We shall then compare the accuracy of the four distinct machine learning algorithms, and try to determine which has the best accuracy.

2. Why machine learning

Machine learning is a type of technology that involves getting a computer to learn from datasets and use this knowledge to make decisions. It does this by following two steps. The first step is the training phase, where the computer is fed data that it can learn from. The second step is testing, where the computer uses the data it has learned from the training phase to make decisions that meet the specific requirements of the application [25].

According to [22], there are three categories of machine learning algorithms: Reinforcement Learning, Supervised Learning, and Unsupervised Learning, as shown in Figure 1.

Reinforcement Learning involves teaching the computer to achieve goals by trial and error and providing rewards for successful outcomes.

Supervised Learning uses labeled data (data belonging to known groups) to understand patterns in the data and then use them to identify and predict data groups in the future. In this paper we shall pay attention mainly, but not exclusively, to the classification methods which belong to Supervised Learning methods. They are divided into five sub-categories: Naïve Bayes classifier, Decision Trees, Support Vector Machines, Random Forest, and K–Nearest Neighbours. In this paper we shall also consider linear regression and Neural Networks, which according to [22] belong to Regression methods.

Unsupervised Learning is the process of creating new information and relationships by sorting unlabeled data into groups and categories, creating associative relationships, and using the Hidden Markov Model to predict future probabilities.

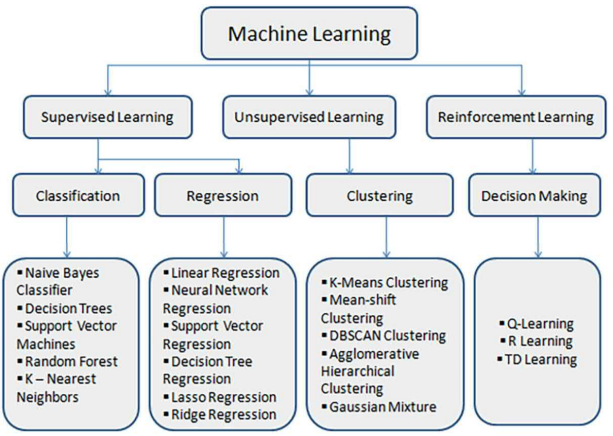


Fig. 1. Machine Learning classification. Source: [22], license: CC BY 3.0.

The machine learning methods seem to be very promising in predicting various heart diseases, which can be concluded from the literature we shall review in the next Section.

3. Literature review

Recently, different studies have been conducted to try and figure out how to predict when someone might have a heart attack. The results of these studies have been published that define proposed solutions to the prediction of Heart Attacks. This section refers to the use of machine learning algorithms, which are sets of instructions to a computer that allow it to learn from data, and about how researchers have used these algorithms to predict heart attacks. Below we have summarized and discussed a number of works to see how successful they have been.

Saw et al. [18] used data from electronic medical records, examining each element of the data and the effects it had on the results. They then created a module to demonstrate the comparison between the data before and after any changes, or *wrangling*, had been made. The authors used logistic regression for analyzing the data and finding patterns in it. Random search was used to look for the best parameters for the prediction model. The patients were classified into two groups: those who have cardiac disease, and those without cardiac disease. The Sklearn Python library (better known under the name Scikit-learn) [1] was used. The accuracy was calculated at 87%, which means that the method was able to correctly predict heart attack risk with a good level of accuracy. The authors used one machine learning algorithm and a small dataset to train and test it,

which could be seen as a disadvantage of this work. A better algorithm could produce better results [12, 24].

The study [25] by Yekkala et al. suggests a way of finding important features in data related to heart disease. It proposes a specific strategy for evaluating which features to pay attention to and then uses the random forest algorithm and rough sets to classify different types of cardiac diseases. The *Heart Disease* database [2] was used, which is a part of the University of California Irvine (UCI) repository [21], a collection of research datasets that can be used to study areas such as computer science and machine learning. This dataset contained 270 individual pieces of data, referred to as *instances*, and was cleaned up before being usedby deleting null or irrelevant instances. After that Exploratory Data Analysis (EDA) was applied the precision of this method touched 84%. The disadvantage of this work is that the parameters used were not introduced, so the algorithms are accurate and effective; however, the results can not be repeated.

Keerthika et al. [6] suggest that machine learning algorithms, such as K-NN, SVM, Logistic Regression, Decision Trees, Random Forests, and Naïve Bayes, can be used to predict heart diseases. These algorithms were applied to *Statlog (Heart)* data [19] from the UCI repository [21].. The result was that the K-NN algorithm achieved a score of 87%, which means it was successful in correctly classifying 87% of the data. The disadvantage of this approach is that it has multiple parameters so it is difficult to set their proper values. To select the values of multiple parameters the grid search or random search could be applied.

Guruprasad et al [4] used data on heart diseases from the *Statlog (Heart)* repository [19]. The data were cleaned. Finally, they used a combination of two different statistical models (a hybrid random forest and a linear model) to try to predict if someone may have heart disease. The proposed method has occurred to have a high accuracy of 92% which is an advantage. However, even though it is considered the classifier with the best accuracy, the results of the proposed method showed that it had the sensitivity of 90%, the lowest one when compared with other algorithms, which is a disadvantage.

Kim et al. [8] proposed a method called NN-FCA. It involved two steps. The first step was feature selection, which involved picking relevant data from the dataset. The second step was feature correlation, where relationships between different variables in the dataset were explored. Then, the Neural Network classification algorithm was used on the KNHANES-VI dataset [9]. This method has the advantage of providing high accuracy when predicting heart diseases. The disadvantage is that the dataset is too small to do effective correlation analysis, so traditional machine learning algorithms can provide similar performance without the added complication.

Kasbe et al. [5] proposed using a fuzzy expert system for predicting heart disease. This system consisted of three major steps: *fuzzification*, *rule base*, and *defuzzification*. The defuzzification step applied the centroid technique and the system used thirteen different input parameters and one output parameter. The dataset used was taken from

Tab. 1. Accuracy of algorithms used in the previous studies.

No.	Algorithm Name	Accuracy $\in [0, 1]$
1	Random Forest	0.94
2	Decision Tree	0.93
3	XGBoost	0.92
4	Support Vector Machine	0.90
5	K-Nearest Neighbour	0.70
6	Logistic Regression	0.69
7	Naïve Bayes	0.92

the UCI repository [19]. The benefit of this system is that it can be used easily and the people who need it can use the system by themselves, while the accuracy is 93.33%. While this system works well, it also adds complexity to the system due to the use of fuzzy logic, and the results do not significantly differ from other systems and studies done on the same dataset.

Shiva et al. [12] used a local dataset toaccurately predict heart attacks. They used a correlation matrix to determine which features were most important and then used three different algorithms: a neural network, SVM and K-NN. The neural network showed the best results, with an accuracy of 93%. The advantage of this method is that the three algorithms are all stable despite using different sizes of the dataset. The downside is that the local dataset used is not representative of the whole global population.

Preetam et al. [16] suggested a hybrid genetic neural network algorithm as a method to speed up predicting heart attacks from ECG signals. The first step before doing this was to pre-process the data, which involves getting rid of any incorrect or extra data and finding patterns in the ECG signals [3]. Then, they used the neural network and connected it to a genetic algorithm to optimize the neural network weights. The benefit of this method is that it can be used to reduce the time needed to predict heart attacks. However, it also brings complexity to the neural network.

Malavika et al. [11] proposed some traditional machine learning algorithms (like Naïve Bayes classifier, Logistic Regression, Random Forest, SVM, Decision Tree Classifier, and K-NN) to predict heart diseases. These algorithms were trained and tested on the UCI Statlog (Heart) dataset [19]. It was shown that the random forest algorithm had the best accuracy. A *pro* of using these traditional algorithms is that 91.17% accuracy was achieved, which is considered an acceptable accuracy. However, a *con* is that the dataset did not use any feature extraction techniques, which could have helped improve the results.

The accuracies achieved with the described methods can be seen Table 1, and the papers together with the databases used in them and the methods used have been gathered in Table 2.

Tab. 2. Summarized papers, datasets and algorithms.

No.	Paper	Year	Dataset	Algorithm
1	[20]	2019	Heart Disease dataset [2]	Logistic Regression
2	[25]	2018	Heart Disease dataset [2]	Random Forest, rough sets
3	[18]	2018	Author's own dataset	Logistic Regression, Naïve Bayes Classifier, Random Forest classifier
4	[6]	2022	Statlog (Heart) dataset [19]	Decision Tree, Language Model, Random Forest, Support Vector Machine,
5	[8]	2017	KNHANES-VI dataset [9]	K-NN, SVM, Logistic Regression, Decision Trees, Random Forest, Naïve Bayes
6	[5]	2017	Statlog (Heart) dataset [19]	Fuzzy Logic
7	[12]	2021	Author's own dataset	Neural Networks, K-NN, SVM
8	[16]	2020	MIT-BIH [3]	Neural Network Model
9	[11]	2020	Statlog (Heart) dataset [19]	Logistic Regression, K-NN, SVM, Naïve Bayes, Decision Tree, Random Forest

4. Methodology

Figure 2 is a visual representation of the system's methodology and illustrates the steps made to analyze patients' data. First, the patients' data are collected and important attributes are selected. Then, the data goes through a pre-processing step which involves data cleaning. The data is then put through a number of algorithms, such as Random Forest, K-Nearest Neighbour, Logistic Regression, Extreme Gradient Boost (XGBoost), and SVM, to perform the training and testing processes and to generate accuracy scores that represent the quality of classification of the heart disease patterns in the data set.

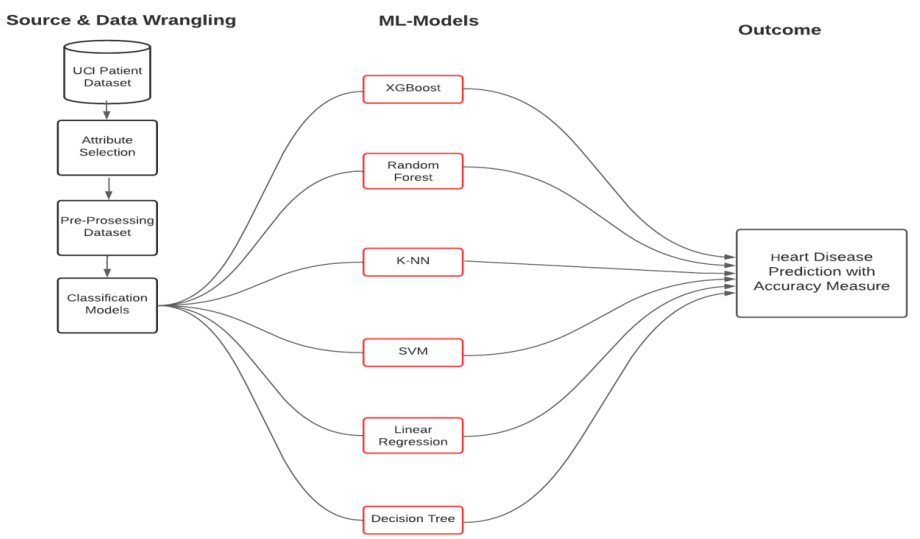
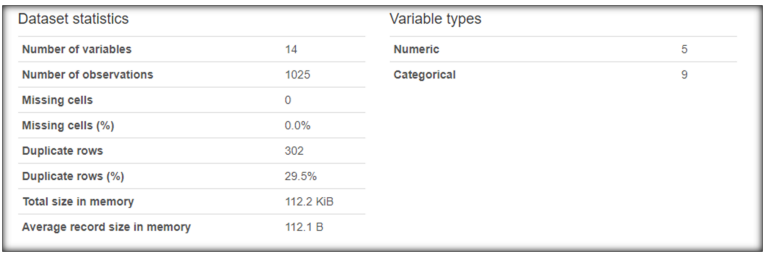


Fig. 2. Methodology used in the construction of the prediction system.



Dataset statistics	
Number of variables	14
Number of observations	1025
Missing cells	0
Missing cells (%)	0.0%
Duplicate rows	302
Duplicate rows (%)	29.5%
Total size in memory	112.2 KiB
Average record size in memory	112.1 B

Variable types	
Numeric	5
Categorical	9

Fig. 3. Overview of the dataset.

4.1. Collection of Data

The UCI repository [21] is a collection of datasets that have been compiled and evaluated by many researchers and UCI authorities. In this paper, we are using the *Statlog (Heart* disease dataset [19] from this repository. We have divided the dataset into two parts, 25% used as test data and 75% used as training data. The the training data will be used to develop the prediction mechanism and the test data will be used to test its effectiveness.

4.2. Feature Engineering

Feature Engineering is the process of choosing important characteristics, or features, of a dataset to create column variables. This means that the characteristics are used in a way that the data can be analysed. The variables will be used as the input data for the prediction system.

One of the methods of feature selection which can be applied without relation to the classification method which will later be used is the $L1$ Regularization (see for example [23]) which involves training a linear model that uses an $L1$ penalty. As a result of the process, the weights of unimportant features in the resulting model are zero. The non-zeroed features are used in the machine-learning model.

In the case of this study, there were 13 features (and one outcome, hence 14 variables) in the database. The number is not large and all the prediction models were capable of capturing this number, so all the features were used in the training and testing. The overview of the dataset and the features are shown in the next Section.

4.3. Overview of the dataset and the variables

The general overview of the dataset used is shown in Fig. 3. The variables, or features, selected for use in the analysis, together with their basic statistical characteristics, are displayed in Fig. 4. The meaning of the abbreviations used in the names of the variables are summarized in Tab. 3.

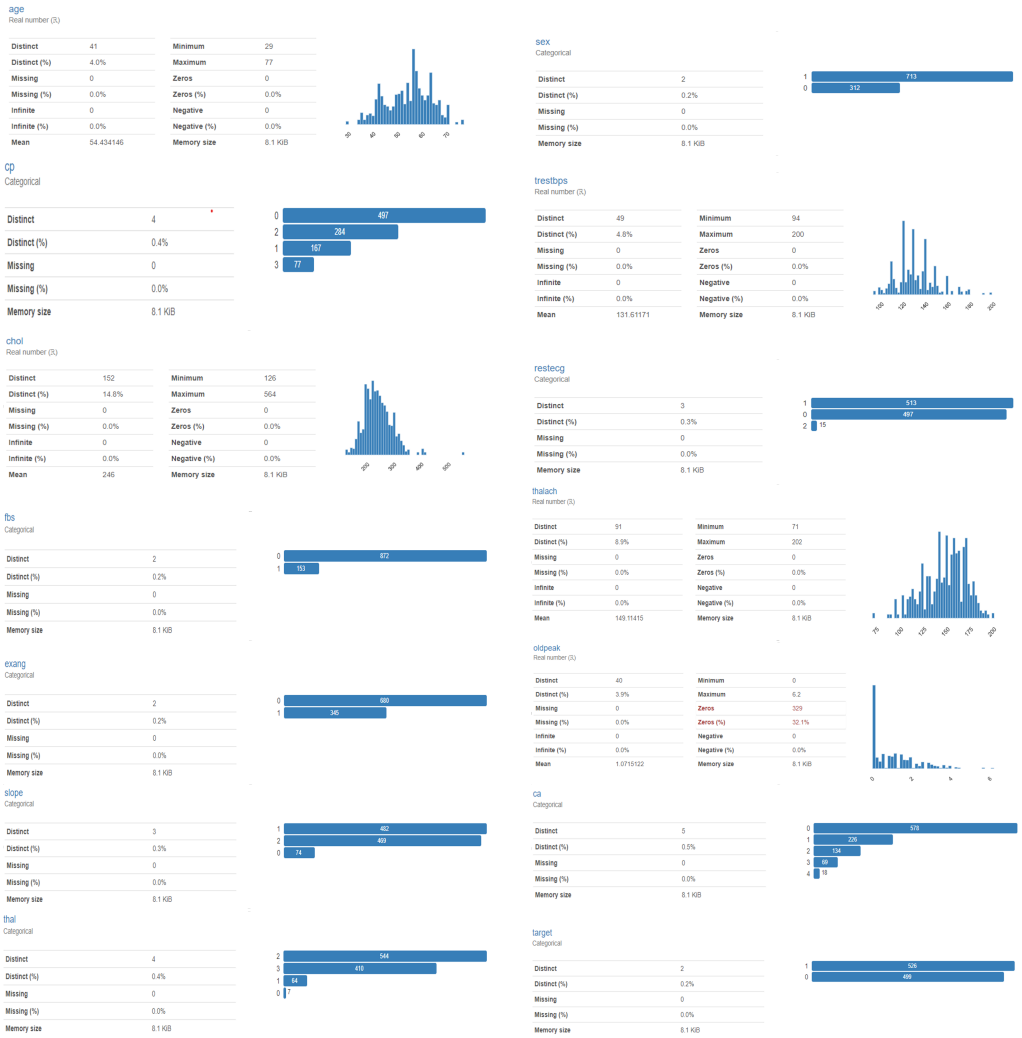


Fig. 4. Variables used in the analysis and their characteristics.

4.4. Correlation

The correlation between the features is shown in Fig. 5 using a heatmap plot. It shows how similar (over zero) or different (below zero) the characteristics of the dataset are. The heatmap was made with the Pandas Profiling tool in Visual Studio Code.

4.5. Performance of Machine Learning algorithms

In this paper several different algorithms were used to build different models which characterize themselves with various levels of accuracy. These algorithms include Random Forest, K-Nearest Neighbour, Logistic Regression, Gradient Boosting, and SVM. The measures of quality of these algorithms were checked on the testing set, to evaluate the performance of these machine learning algorithms for predicting the likelihood of heart attacks.. In the following, the results for each model are described in more detail.

Specifically, the Random Forest model achieved an AUC score of 0.9887, indicating its ability to discern patterns and make accurate predictions. Similarly, the K-Nearest Neighbour model exhibited a commendable AUC score of 0.9468, highlighting its predictive capabilities.

We also assessed the Logistic Regression model, which yielded an AUC score of 0.9391. Although slightly lower than the other models, it still demonstrated a valuable predictive capacity. Furthermore, the Extreme Gradient Boost model exhibited a strong performance, with an AUC score of 0.9774, indicating its ability to leverage boosting techniques and generate accurate predictions.

Notably, the Support Vector Machine (SVM) model stood out with an exceptional AUC score of 0.9985. This result showcases the SVM's robustness in accurately classifying individuals as either at risk or not at risk of heart attacks. The SVM model's ability to leverage kernel functions and identify complex patterns within the dataset contributes to its remarkable predictive accuracy.

Tab. 3. The meaning of the abbreviations used in the names of the variables (simplified nonmedical description).

No.	Abbrev.	Meaning	No.	Abbrev.	Meaning
1	age	age	2	sex	sex
3	cp	chest pain	4	trestbps	resting blood pressure
5	chol	cholesterol level	6	restecg	resting electrocardiographic measurement (0: normal, 1: ST-T wave abnormality, 2: left ventricular hypertrophy)
7	fbs	fasting blood sugar	8	thalach	maximum heart rate achieved
9	exang	exercise induced angina	10	oldpeak	ST depression induced by exercise relative to rest
11	slope	ST segment shift relative to exercise-induced increments in heart rate	12	ca	number of major vessels
13	thal	thalassemia	14	target	heart disease, study target

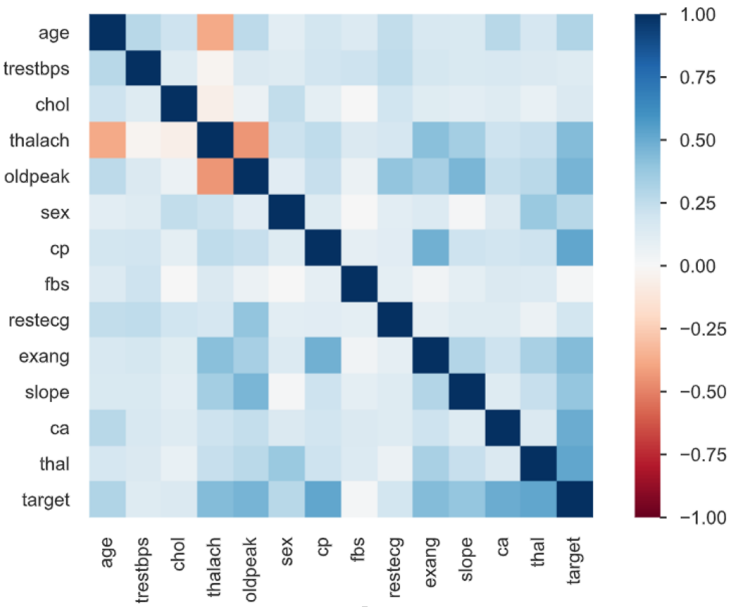


Fig. 5. Heatmap of the dataset features.

The high AUC scores obtained by these machine learning models highlight their effectiveness in predicting the likelihood of heart attacks. These findings have significant implications for early detection and prevention strategies in cardiovascular health. By incorporating these models into clinical practice, healthcare professionals can enhance risk assessment and provide personalized interventions to individuals at higher risk.

Overall, our study demonstrates the potential of machine learning algorithms, including Random Forest, K-Nearest Neighbour, Logistic Regression, Extreme Gradient Boost, and Support Vector Machine, in accurately predicting the likelihood of heart attacks. These models can assist healthcare professionals in identifying individuals who may benefit from targeted preventive measures, ultimately contributing to improved cardiovascular health outcomes.

5. Software used

In this study the Visual Studio Code environment was used as the programming platform, with its support for C++, C#, .NET and its ability to cooperate with many different types of languages and tools (such as JavaScript, Type Script, Node.js, Java, Python,

PHP, and others). In the calculations and in producing the graphs shown in this paper the NumPy and PyTorch packages [17] in Python ver. 3.10 were used.

6. Result analysis

6.1. Accuracy and other measures of performance quality

The confusion matrix is a visual representation of how well a classifier, or a model for predicting outcomes, performs on a set of test data. It compares the actual results of the model to the expected outcomes and is useful for understanding the results.

The accuracy of algorithms is calculated based on four values: true negative (TN) – the number of people without heart diseases who have been identified correctly, true positive (TP) – the number of people with heart diseases who have been identified correctly, false positive (FP) – the number of people without heart disease who have been incorrectly identified as having heart disease, and false negative (FN) – the number of people with heart disease who have been incorrectly identified as being healthy.

The general measure of decision quality taking into account the true as well the false results, and the positive and negative ones, is the accuracy:

$$\mathrm{Acc} = \frac{\mathrm{TP} + \mathrm{FN}}{\mathrm{TP} + \mathrm{FP} + \mathrm{TN} + \mathrm{FN}} \,.$$

There are at least two detailed aspects of a classifier (we shall stay with the detection of disease as an example). These are: its sensitivity – ability to classify the patients with a disease as ill, and its sensitivity – ability not to classify the healthy patients as ill.

$$\begin{aligned} \mathrm{Sens} &= \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}} \,, \\ \mathrm{Spec} &= \frac{\mathrm{TN}}{\mathrm{FP} + \mathrm{TN}} \,. \end{aligned}$$

The general performance of the system can be more accurately described by the confusion matrix containing all the four numbers, as shown in Fig. 6.

The single parameter which captures the performance of a classification system is the Area Under Curve (AUC). It is related to the Receiver Operating Characteristics (ROC) which is a curve formed by all the pairs of values of the sensitivity and specificity, for all possible thresholds in the classifier. The best value of the AUC is 1. For details please see the basic literature [10, 14].

6.1.1. Performance of Machine Learning Algorithms

To compare the accuracy scores between those received in the previous studies, summarized in Table 1, and those achieved in the present study, shown in Table 4, the results of the subsequent algorithms will be presented.

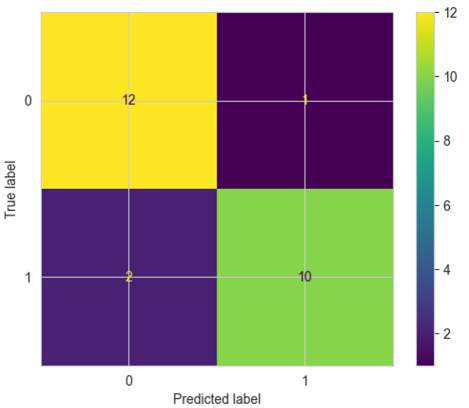


Fig. 6. Example of a confusion matrix of a trained model.

Random Forest

In Table 1 (previous studies), the Random Forest algorithm achieved an accuracy of 0.94, while in Table 4 (current study), it achieved a slightly lower accuracy of 0.93. This indicates a minor variation in the performance of the Random Forest algorithm between the previous studies and the current study.

Decision Tree

Table 1 (previous) shows a Decision Tree accuracy of 0.93, whereas Table 4 (current) reports an accuracy of 0.94. This suggests that the Decision Tree algorithm performed slightly better in the current study compared to the previous studies.

Tab. 4. Accuracy and Area Under Curve (AUC) of the methods used in this study.

No.	Model	Accuracy [%]	AUC $\in [0,1]$
1	Logistic Regression	86.34	0.939
2	Naïve Bayes	85.37	0.931
3	Random Forest	93.67	0.989
4	Extreme Gradient Boost	94.64	0.977
5	K-Nearest Neighbour	87.81	0.947
6	Decision Tree	94.64	1.000
7	Support Vector Machine	98.05	0.998

Extreme Gradient Boost)
In Table 1 (previous), XGBoost achieved accuracy of 0.92, and in Table 4 (current), XGBoost achieved an accuracy of 0.94. This indicates an improvement in the performance of the XGBoost algorithm in the current study compared to the previous studies.

Support Vector Machine
In Table 1 (previous), Support Vector Machine achieved an accuracy of 0.90, while in Table 4 (current) it achieved a higher accuracy of 0.98. This indicates a significant improvement in the performance of the Support Vector Machine algorithm in the current study compared to the previous studies.

K-Nearest Neighbour
Table 1 (previous) shows an accuracy of 0.70 for K-Nearest Neighbour, whereas Table 4 (current) reports a higher accuracy of 0.87. This suggests that the K-Nearest Neighbour algorithm performed better in the current study compared to the previous studies.

Logistic Regression
Table 1 (previous) indicates a Logistic Regression accuracy of 0.69, while Table 4 (current) reports a higher accuracy of 0.86. This indicates an improvement in the performance of the Logistic Regression algorithm in the current study compared to the previous studies.

Naïve Bayes
In Table 1 (previous), Naïve Bayes achieved an accuracy of 0.922, while in Table 4 (current) it achieved a lower accuracy of 0.85. This suggests a slight decrease in the performance of the Naïve Bayes algorithm between the previous studies and the current study.

7. Discussion of results

In our study, we evaluated the performance of several machine learning algorithms for predicting the likelihood of heart attacks, as shown in Tab. 4. The tested algorithms were: Logistic Regression, Naïve Bayes, Random Forest, Extreme Gradient Boost, K-Nearest Neighbour, Decision Tree, and Support Vector Machine. We found that SVM had the highest accuracy.

The results demonstrated the effectiveness of these models in accurately classifying individuals at risk of heart attacks. Specifically, the Random Forest model achieved an AUC score of 0.989, indicating its ability to discern patterns and make accurate predictions. Similarly, the K-Nearest Neighbour model exhibited a commendable AUC score of 0.947, highlighting its predictive capabilities.

We also assessed the Logistic Regression model, which yielded an AUC score of 0.939. Although slightly lower than the other models, it still demonstrated a valuable

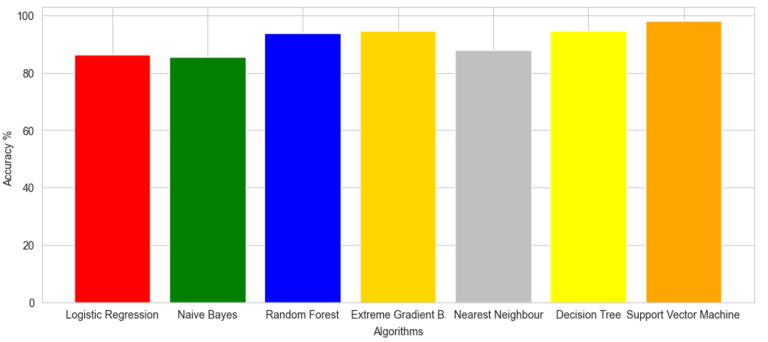


Fig. 7. Accuracy of the Machine Learning models tested in this study.

predictive capacity. Furthermore, the Extreme Gradient Boost model exhibited a strong performance, with an AUC score of 0.977, indicating its ability to leverage boosting techniques and generate accurate predictions.

Notably, the Support Vector Machine (SVM) model stood out with an exceptional AUC score of 0.998. This result showcases the SVM's robustness in accurately classifying individuals as either at risk or not at risk of heart attacks. The SVM model's ability to leverage kernel functions and identify complex patterns within the dataset contributes to its remarkable predictive accuracy.

The accuracies of the algorithms are compared graphically in Figure 7

The high AUC scores obtained by these machine learning models highlight their effectiveness in predicting the likelihood of heart attacks. These findings have significant implications for early detection and prevention strategies in cardiovascular health. By incorporating these models into clinical practice, healthcare professionals can enhance risk assessment and provide personalized interventions to individuals at higher risk. Overall, our study demonstrates the potential of machine learning algorithms, including Random Forest, K-Nearest Neighbour, Logistic Regression, Extreme Gradient Boost, and Support Vector Machine, in accurately predicting the likelihood of heart attacks. These models can assist healthcare professionals in identifying individuals who may benefit from targeted preventive measures, ultimately contributing to improved cardiovascular health outcomes.

8. Conclusion and future work

The heart is an essential part of the body. To predict cardiac issues, machine learning algorithms are needed to help treat these cardiac ailments. This research paper used seven different machine-learning algorithms to predict heart disease and found that the

Support Vector Machine algorithm was the most accurate. We compared the results of different machine learning algorithms to determine the one that is most suitable for predicting heart disease to predict whether a person has heart disease or not. In this research paper, accuracy is an important factor that is used to measure how well the algorithm works. The dataset used for the research contains 14 attributes (13 features and one target). Out of the seven machine learning algorithms tested to predict heart disease, the Support Vector Machine was found to be the most accurate prediction ability on a UCI dataset. In the future, more machine learning techniques could be used to analyse cardiac illnesses and predict them earlier. With the help of proper machine learning technologies, we can hopefully have a lower number of fatalities related to heart problems.

References

[1] D. Cournapeau, M. Brucher, F. Pedregosa, et al. scikit-learn. Machine Learning in Python, 2023. `https://scikit-learn.org`.

[2] R. Detrano, A. Jánosi, W. Steinbrunn, et al. Heart Disease. In *UC Irvine Machine Learning Repository*. 1988. doi:10.24432/C52P4X.

[3] G. Moody and R. Mark. MIT-BIH Arrhythmia Database, 2005. doi:10.13026/C2F305.

[4] S. Guruprasad, V. L. Mathias, and W. Dcunha. Heart disease prediction using machine learning techniques. In *Proc. 2021 5th Int. Conf. Electrical, Electronics, Communication, Computer and Optimization Techniques (ICEECCOT)*, pages 762–766, Mysuru, India, 10-11 Dec 2021. IEEE. doi:10.1109/ICEECCOT52851.2021.9707966.

[5] T. Kasbe and R. S. Pippal. Design of heart disease diagnosis system using fuzzy logic. In *Proc. 2017 Int. Conf. Energy, Communication, Data Analytics and Soft Computing (ICECDS)*, pages 3183–3187, Chennai, India, 01-02 Aug 2017. IEEE. doi:10.1109/ICECDS.2017.8390044.

[6] N. V. S. Keerthika Dulam, K. Sai Koushik, S. Sakhamuri, Ch. Hyndavi, and G. Sindu. Heart disease prediction with machine learning approaches. In *Proc. 4th Int. Conf. Communications and Cyber Physical Engineering (ICCCE 2021)*, volume 828 of *Lecture Notes in Electrical Engineering*, pages 1155–1166. Springer, 2022. doi:10.1007/978-981-16-7985-8_120.

[7] S. Khan, S. M. D. Rizvi, V. Ahmad, et al. Magnetic nanoparticles: properties, synthesis and biomedical applications. *Current Drug Metabolism*, 16(8):685–704, 2015. `https://www.ingentaconnect.com/content/ben/cdm/2015/00000016/00000008/art00009`.

[8] J. K. Kim and S. Kang. Neural network-based coronary heart disease risk prediction using feature correlation analysis. *Journal of Healthcare Engineering*, 2017:2780501, 2017. doi:10.1155/2017/2780501.

[9] Korea Disease Control and Prevention Agency. Korea National Health and Nutrition Examination Survey (KNHANES), 2014-present. `https://knhanes.kdca.go.kr`.

[10] L. B. Lusted. Signal detectability and medical decision-making. *Science*, 171:1217–1219, 1971. doi:10.1126/science.171.3977.1217.

[11] G. Malavika, N. Rajathi, V. Vanitha, and P. Parameswari. Heart disease prediction using machine learning algorithms. *Bioscience Biotechnology Research Communications*, 13(11):24–27, 2020. Special Issue. doi:10.21786/bbrc/13.11/6.

[12] B. Shiva Shanta Mani and V. M. Manikandan. Heart disease prediction using machine learning. In G. Rani and P. K. Tiwari, editors, *Handbook of Research on Disease Prediction Through Data Analytics and Machine Learning*, chapter 18, page 373–381. IGI Global, 2021. doi:10.4018/978-1-7998-2742-9.ch018.

[13] J. Maurya and S. Prakash. Machine learning based prediction and diagnosis of heart disease using multiple models. *ResearchSquare*, 2023. Preprint. doi:10.21203/rs.3.rs-2642516/v1.

[14] C. E. Metz. Basic principles of ROC analysis. *Seminars in Nuclear Medicine*, 8(4):283–298, 1978. doi:10.1016/S0001-2998(78)80014-2.

[15] P. Ponikowski, S. D. Anker, K. F. AlHabib, et al. Heart failure: preventing disease and death worldwide. *ESC heart failure*, 1(1):4–25, 2014. doi:10.1002/ehf2.12005.

[16] T. V. N. Preetam, Dr P. Kshirsagar, M. V. Krishna, and V. Meghana. ECG signal analysis and prediction of heart attack with the help of optimized neural network. *Alochana Chakra Journal*, 9:497–506, 2020.

[17] Python Software Foundation. The Python Package Index (PyPI), 2023. `https://pypi.org/`.

[18] M. Saw, T. Saxena, S. Kaithwas, R. Yadav, and N. Lal. Estimation of prediction for getting heart disease using logistic regression model of machine learning. In *Proc. 2020 Int. Conf. Computer Communication and Informatics (ICCCI)*, pages 1–6, Coimbatore, India, 22-24 Jan 2020. IEEE. doi:10.1109/ICCCI48352.2020.9104210.

[19] Statlog (Heart). In *UC Irvine Machine Learning Repository*. doi:10.24432/C57303.

[20] K. Subhadra and B. Vikas. Neural network based intelligent system for predicting heart disease. *International Journal of Innovative Technology and Exploring Engineering*, 8(5):484–487, 2019. `https://www.ijitee.org/portfolio-item/D2770028419/`.

[21] UC Irvine Machine Learning Repository, 2023. `https://archive.ics.uci.edu`.

[22] K. Ukoba and T.-C. Jen. Biochar and application of machine learning: A review. In M. Bartoli, M. Giorcelli, and A. Tagliaferro, editors, *Biochar – Productive Technologies, Properties and Applications*, chapter 14. IntechOpen, Rijeka, 2022. doi:10.5772/intechopen.108024.

[23] Diego Vidaurre, Concha Bielza, and Pedro Larrañaga. A survey of L1 regression. *International Statistical Review*, 81(3):361–387, 2013. doi:10.1111/insr.12023.

[24] A. S. Wale, S. S. Sonawani, and S. C. Karande. ECG signal analysis and prediction of heart attack with the help of optimized neural network using Genetic Algorithm. In *Proc. Int. Conf. Emerging Trends in Engineering, Technology, Science and Management*, Greater Noida, India, 2017. IIMT College of Engineering.

[25] I. Yekkala and S. Dixit. Prediction of heart disease using random forest and rough set based feature selection. *International Journal of Big Data and Analytics in Healthcare*, 3(1):1–12, 2018. doi:10.4018/IJBDAH.2018010101.