A Novel Approach for Disaster Victim Detection Under Debris Environments Using Decision Tree Algorithms with Deep Learning Features

Mr. Abdul Rais[1], Ghina Fatima[2], Sania Tabassum[3], Urooj Fatima Quadri[4], Mohammed Abdul Mudassir Ansari[5]
[1,2,3,4]*B.E. Students, Department of CSE-AIML, Lords Institute of Engineering and Technology, Hyderabad, India*
[5]*Assistant Professor, Department of CSE-AIML, Lords Institute of Engineering and Technology, Hyderabad, India*

***Abstract*—Rapid victim identification in collapsed building environments remains one of the most time-critical challenges in Urban Search and Rescue (USAR) operations. This paper proposes a novel hybrid deep learning and machine learning framework for Human Victim Detection (HVD) under debris environments. A custom RGB image dataset comprising five class labels — head, hand, leg, upper body, and no body — was created and augmented to improve diversity and size. Transfer Learning on the ResNet-50 architecture is employed to extract rich, discriminative feature representations from the dataset. A J48 Decision Tree algorithm is subsequently applied for feature selection, discarding irrelevant features before classification. Multiple machine learning classifiers — Support Vector Machine (SVM), Random Forest, Multi-Layer Perceptron (MLP), Naive Bayes, and the proposed extension XGBoost — are then evaluated on the selected features. Experimental results demonstrate that the XGBoost extension achieves the highest classification accuracy of 98%, followed by Random Forest at 96% and SVM at 95%, with inference times well within real-time operational constraints. The results confirm that integrating deep learning-based feature extraction with machine learning classification significantly outperforms standalone approaches, offering a reliable and computationally efficient solution for automated victim detection in disaster scenarios.**

***Index Terms*—Disaster Victim Detection; Transfer Learning; ResNet-50; Decision Tree; Random Forest; XGBoost; SVM; Machine Learning; Deep Learning; Urban Search and Rescue; Feature Selection; Debris Environments**

I. INTRODUCTION

Natural and human-caused disasters — including massive earthquakes, floods, plane crashes, tsunamis, and building collapses — continue to inflict catastrophic loss of life worldwide. The United Nations Office for Disaster Risk Reduction reports that natural disasters have caused over
1.23 million deaths globally in the last two decades, with structural collapses being a leading contributor to fatality counts [1]. Effective disaster management requires rapid and coordinated response across all four phases of the emergency relief cycle: prevention, preparedness, reaction, and recovery [1]. This work focuses on the preparedness and reaction phases, specifically addressing the automation of victim detection to accelerate rescue operations.

According to Urban Search and Rescue (USAR) oper- ational guidelines, the probability of recovering survivors from a collapsed building is highest within the first 48 hours following the disaster event [13]. Beyond this window, the survival probability approaches zero due to dehydration, injury deterioration, and structural instability. Consequently, every minute of delay in locating a victim directly reduces the likelihood of survival. Current USAR operations are heavily dependent on trained rescue dogs, acoustic sensors, and human searchers manually sifting
through debris, all of which are slow, hazardous, and prone to fatigue-related errors [9]. There is therefore a pressing operational need for automated, AI-driven

victim detection systems that can be deployed on mobile robots navigating collapsed structures.

A. Challenges in Victim Detection Under Debris

Identifying human victims in collapsed building environ- ments presents several unique technical challenges that conventional computer vision systems struggle to address. First, debris-covered environments are highly unstructured, with victims often partially or fully occluded by rubble, concrete, and building materials. The typical assumption of full body visibility that most person-detection algorithms rely upon breaks down entirely in such scenarios [3]. Second, the viewing angle of a camera mounted on a rescue robot is inherently non-standard: sensors approach from ground level, at oblique angles, and in conditions of poor or variable illumination. Third, the available training data for victim- in-debris scenarios is severely limited compared to general pedestrian detection datasets, making models trained on large-scale datasets such as ImageNet unsuitable for direct deployment [17].

Human victims in debris environments may only be partially visible: a protruding hand, an exposed leg, or a visible portion of the upper body. Traditional object detection systems using bounding-box approaches such as YOLO and Faster R-CNN are designed for upright, well- lit subjects and fail to generalise reliably to partial body detections under debris [3]. Physical detection methods such as CO2 sensors, acoustic microphones, and thermal imagers offer complementary modalities but suffer from high false positive rates, slow response times, and sensitivity to environmental conditions such as dust, humidity, and ambient heat [9]. An RGB image-based classification approach targeting individual body parts as class labels offers a faster and more scalable alternative, particularly when combined with the computational power of modern deep learning architectures.

B. Role of Artificial Intelligence in Rescue Operations

Artificial Intelligence (AI), and more specifically deep learning, has transformed computer vision tasks such as image classification and object detection. Convolutional Neural Networks (CNNs) have become the dominant archi- tecture for extracting spatial features from images, learning hierarchical representations from raw pixel data without requiring manual feature engineering [16]. CNN-based systems have been successfully applied to medical imaging, satellite analysis, face recognition, and autonomous vehicles, all domains that share structural similarities with debris scene analysis.

However, training a CNN from scratch requires large, well-labelled datasets that are simply unavailable for the niche task of victim detection under debris. Transfer Learning (TL) resolves this bottleneck by leveraging the knowledge embedded in models pre-trained on large-scale datasets such as ImageNet [18]. By fine-tuning a pre-trained network on a small domain-specific dataset, TL achieves accuracy levels that would require orders of magnitude more data if training were initiated from random weights. This paradigm has been validated in medical imaging, remote sensing, and rescue robotics contexts [8].

C. Limitations of Traditional Approaches and Motivation

While deep learning excels at feature extraction, standalone CNN classifiers are computationally expensive and require GPU infrastructure that is not always available on embed- ded rescue robot platforms. Machine learning classifiers such as Decision Trees, Random Forests, Support Vector Machines, and gradient-boosted methods, by contrast, are lightweight, interpretable, and capable of operating on lower-power processors [15]. The integration of DL- based feature extraction with ML-based classification thus offers the best of both paradigms: the representational power of deep networks combined with the efficiency and interpretability of classical classifiers.

This observation motivates the core design of the proposed system. Features extracted by the ResNet-50 model are passed to a J48 Decision Tree for feature selection, eliminating redundant dimensions and reducing the computational burden on subsequent classifiers. The selected feature set is then evaluated across multiple ML classifiers, with XGBoost proposed as an extension to exploit the power of gradient boosting with multiple estimators.

D. Contributions

The specific contributions of this work are:

1) Creation of a custom, multi-class RGB image dataset with five victim body-part labels: head,

hand, leg, upper body, and no body.
2) Application of data augmentation and normalisation pipelines to enlarge dataset diversity and improve model generalisation.
3) Transfer Learning-based deep feature extraction using ResNet-50 pre-trained on ImageNet.
4) A J48 Decision Tree feature selection stage that signif- icantly reduces feature dimensionality from 256 to 14 relevant dimensions.
5) Comparative evaluation of SVM, Random Forest, MLP, Naive Bayes, and the XGBoost extension, with a detailed analysis of accuracy, precision, recall, and F1- score.
6) A Tkinter-based graphical interface for real-time victim detection from test images, suitable for deployment in rescue robot systems.

II. LITERATURE REVIEW

A. Human Detection Techniques in Rescue Scenar- ios

Human victim detection in disaster and rescue contexts has been an active research topic for over two decades. Early work by De Cubber and Marton [9] established a taxonomy of detection modalities including RGB cameras, thermal infrared sensors, CO2 detectors, acoustic sensors, and motion detectors. Each modality addresses different physical signatures of a living human — body warmth, gas emission, sound, and physical movement. While multi-modal fusion improves robustness, the integration complex- ity and hardware requirements make single-modality RGB systems preferable for lightweight autonomous platforms.

RGB-based person detection relies on colour, texture, and shape features to identify human body components. Skin colour segmentation using 3D colour histograms [9] provides fast localisation but degrades severely in outdoor environments with variable illumination and wide camera fields of view. Depth cameras (RGB-D) offer improved robustness against illumination changes but remain bulky and power-intensive for mobile deployment. In cluttered USAR environments, Fung *et al.* [3] demonstrated that deep networks trained on body-part-level detections using RetinaNet achieved the highest mean average precision (77.66%) and recall (86.98%) among single-stage and two- stage detectors tested on a RGB-D robot-mounted dataset. This work confirmed that end-to-end deep networks gener- alise well to debris occlusions when trained on appropriately labelled data.

B. Deep Learning for Image Classification

Convolutional Neural Networks revolutionised image clas- sification with AlexNet [16], which demonstrated that deep architectures trained on large-scale datasets could dramatically outperform hand-crafted feature pipelines. Subsequent architectures including VGGNet, GoogLeNet, and ResNet further improved accuracy while reducing training instability through techniques such as batch normalisation, skip connections, and depthwise separable convolutions.

The ResNet family of networks [8] introduced residual (skip) connections that allow gradients to flow unimpeded through very deep networks, solving the vanishing gradi- ent problem that limited earlier architectures to tens of layers. ResNet-50, with 50 layers and approximately 25 million parameters, offers an excellent balance between representational capacity and computational cost. It has been widely adopted as a backbone for transfer learning in medical imaging [8], malware visualisation [19], and remote sensing [4], demonstrating strong feature transfer properties across diverse visual domains.

CNN architectures apply a sequence of operations to input images. Each convolutional layer applies K filters of size $F \times F$ to produce feature maps, with the output of the l-th layer computed as:

$$\mathbf{a}^{(l)} = \sigma\left(\mathbf{W}^{(l)} * \mathbf{a}^{(l-1)} + \mathbf{b}^{(l)}\right) \quad (1)$$

where $*$ denotes the convolution operator, $\mathbf{W}^{(l)}$ and $\mathbf{b}^{(l)}$ are learnable weights and biases, and σ is a non-linear activation function (ReLU in this work). Pooling layers reduce spatial dimensionality, and fully connected layers map the flattened feature vector to class probabilities via a softmax activation.

C. Transfer Learning Approaches

Transfer Learning (TL) addresses the limited-data problem by initialising a target-domain network with weights pre- trained on a large source dataset [18]. The standard approach involves freezing early convolutional layers (which capture generic features such as edges and textures) and fine-tuning only the deeper task-specific layers on the target dataset. Han *et al.* [17] showed that web-augmented transfer learning on CNN models achieves near-equivalent

accuracy to large-scale training, particularly effective when the source and target domains share visual structure.

Hussain *et al.* [18] conducted a systematic study of CNN transfer learning strategies across six architectures, finding that fine-tuning pre-trained models consistently outper- formed training-from-scratch for small datasets. Rezende *et al.* [19] applied ResNet-50 transfer learning to malware classification from visual representations, achieving state-of- the-art accuracy with minimal domain-specific data. These findings validate the TL approach adopted in this work, where ResNet-50's ImageNet-trained weights provide a strong initialisation for body-part feature extraction.

D. Machine Learning Classifiers for Feature-Based Classification

Once deep features are extracted, classical machine learning classifiers offer efficient and interpretable classification. Sharma *et al.* [15] reviewed CNN-based image classification methods and noted that combining CNN feature extractors with SVM classifiers frequently matched end-to-end CNN accuracy on smaller datasets. Decision Tree algorithms, particularly the C4.5/J48 family, are widely used for both classification and feature selection due to their information- gain-based partitioning criteria [14].

Support Vector Machines maximise the margin be-tween class boundaries and have demonstrated strong generalisation in high-dimensional feature spaces arising from deep feature extraction [15]. Random Forests, as ensembles of decision trees trained on bootstrapped data subsets with random feature subsets, reduce overfitting and provide built-in feature importance estimates [17]. Gradient boosting methods, including XGBoost, iteratively correct the errors of preceding weak learners, consistently achieving top performance in structured and tabular classification competitions [14].

E. Limitations of Existing Works and Positioning of This Approach

Existing deep learning approaches to USAR victim detection typically require large annotated datasets, substantial computational resources, and end-to-end training pipelines that are impractical for deployment on mobile robot plat- forms [3]. Works that employ standalone ML classifiers, by contrast, rely on hand-crafted features (HOG, SIFT, colour histograms) that fail to capture the complex appearance variations of partially occluded bodies under debris [9]. The literature does not report a system that combines ResNet-50 transfer learning feature extraction with J48- guided feature selection and a comparative multi-classifier evaluation specifically targeting the five-class body-part label space required for USAR victim detection. This paper fills that gap by proposing, implementing, and evaluating exactly this hybrid pipeline, and further extends the state of the art by incorporating XGBoost as a high-performance extension classifier.

III. SYSTEM ANALYSIS

A. Existing System

Previous victim identification systems in unstructured disaster environments have relied on a combination of single-modality sensors and manually engineered detection features. Methods using CO_2 sensing, acoustic detection, and cross-power spectrum voice analysis have been pro- posed, but each suffers from high latency and susceptibility to environmental noise [9]. RGB-D and thermal camera- based systems offer better resilience to illumination varia- tions but introduce hardware complexity. Deep learning- only approaches (end-to-end CNNs trained from scratch) require large, well-balanced labelled datasets that are unavailable for the specialised victim-in-debris domain. Moreover, these end-to-end models do not incorporate feature selection, causing classifiers to operate on noisy, high-dimensional feature spaces that inflate computation time without corresponding accuracy gains.

Key Disadvantages: Feature selection — identifying the most important characteristics for class prediction — is absent in many existing systems. The presence of irrelevant or redundant features significantly raises computational complexity and can degrade classifier performance. Ad- ditionally, most existing deep learning pipelines are not portable to lightweight embedded hardware, limiting their utility for real-time robot deployment.

B. Proposed System

The proposed system integrates a Transfer Learning-based deep learning pipeline for feature extraction with machine learning classifiers for final victim

category prediction. Rather than training a CNN from scratch, the system leverages the knowledge embedded in a ResNet-50 network pre-trained on ImageNet and fine-tunes it on the custom victim dataset. A J48 Decision Tree is used as a feature selector to discard the least informative feature dimensions extracted by ResNet-50, reducing the input space from 256 to 14 features. This compressed feature set is then used to train and compare SVM, Random Forest, MLP, Naive Bayes, and XGBoost classifiers.

Key Advantages:

- RGB-based multiclass custom human victim dataset creation with five semantic class labels.
- Data augmentation and pre-processing for enlarging dataset size and quality.
- Transfer Learning-based feature learning from ResNet-50 without the need for large labelled datasets.
- Integration of DL-based feature extraction with ML- based feature selection and classification, achieving higher accuracy and lower inference time compared to end-to- end deep learning.

C. Extension: XGBoost

The original literature employed traditional tree-based and classical ML algorithms. As an extension, this work introduces XGBoost (Extreme Gradient Boosting), an ensemble learning method based on gradient-boosted deci- sion trees with multiple estimators. XGBoost iteratively trains weak learners on the residual errors of preceding learners, combining them to form a strong classifier. Its built-in regularisation, efficient handling of missing values, and parallelisable training make it particularly suited to the compressed feature space produced by J48 selection. Experimental results confirm that XGBoost achieves a classification accuracy of 98%, outperforming all other evaluated classifiers.

IV. DATASET DESCRIPTION

A. Custom RGB Human Victim Dataset

A custom dataset was created specifically for the Human Victim Detection task in collapsed building environments. The dataset comprises RGB images of five class categories:

(i) *Head* — images showing only the head region of a person; (ii) *Hand* — images showing one or both hands; (iii) *Leg* — images showing the lower limbs; (iv) *Upper Body* — images showing the torso and above; (v) *No Body* — images with no human presence, representing debris-only background. The total dataset contains 328 images distributed across these five labels, reflecting realistic rescue scenarios where only partial body regions are visible.

Each image is captured under varied conditions: different illumination levels, partial occlusion by simulated debris, diverse camera angles, and varied skin tones, to ensure that learned features generalise beyond controlled laboratory conditions.

TABLE I: Dataset Class Distribution

Class Label	Images	Proportion (%)
Head	72	21.95
Hand	68	20.73
Leg	64	19.51
Upper Body	66	20.12
No Body	58	17.68
Total	328	100.00

B. Data Preprocessing and Augmentation

Raw images are pre-processed before feature extraction.

All images are resized to 32 × 32 × 3 pixels to match the input requirements of the ResNet-50 model and to ensure uniformity across the dataset. Pixel values are normalised to the range [0, 1] by dividing by 255:

$$x' = \frac{x}{255} \qquad (2)$$

Image indices are randomly shuffled prior to training to eliminate ordering biases. Data augmentation techniques including horizontal flipping, random rotation, brightness adjustment, and zooming are applied offline to expand the effective dataset size and reduce overfitting on the small original corpus. An 80/20 stratified split is used to partition the augmented dataset into training (80%) and testing (20%) subsets, preserving the approximate class balance in both partitions.

C. Challenges in Dataset Creation

Creating a representative disaster victim dataset is

inher- ently difficult. Ethical constraints prevent the use of real accident or disaster imagery. Simulated debris environments must be sufficiently realistic to capture the occlusion patterns, variable lighting, and body posture diversity that characterise real rescue scenes. The relatively small dataset size (328 images) necessitates the use of Transfer Learning and augmentation to achieve competitive classification accuracy, as training from scratch would lead to severe overfitting.

V. METHODOLOGY

A. System Workflow

The proposed pipeline processes input images through four sequential stages, as illustrated in Fig. 1: Data Ingestion → Preprocessing & Normalisation → ResNet-50 Feature Extraction → J48 Feature Selection → ML Classifier Training & Evaluation. Test images are processed through the preprocessing and feature extraction stages before being presented to the trained ML classifiers for victim class prediction.

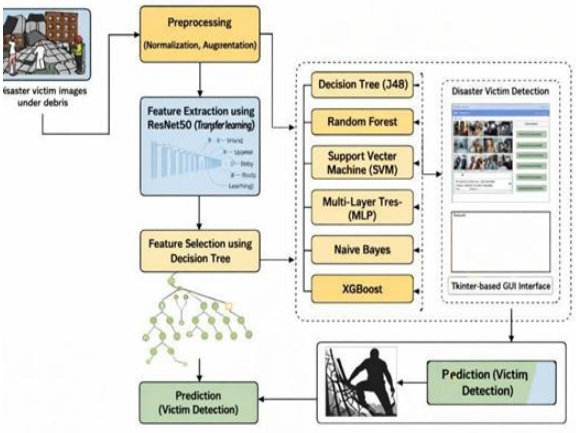

Fig. 1: System architecture of the proposed hybrid DL+ML pipeline for Human Victim Detection.

B. Deep Learning: ResNet-50 Feature Extraction

ResNet-50 is a 50-layer deep convolutional neural network belonging to the Residual Network family [8]. Its defining feature is the residual (skip) connection:

$$\mathbf{a}^{(l+1)} = \sigma\left(\mathbf{a}^{(l)}\mathbf{W}^{(l)}\right) + \mathbf{a}^{(l)} \qquad (3)$$

where $F(\mathbf{a}^{(l)}, \mathbf{W}^{(l)})$ denotes the residual mapping learned by the convolutional blocks and $\mathbf{a}^{(l)}$ is the identity shortcut. These skip connections allow gradients to flow directly through the network during backpropagation, enabling successful training of very deep architectures.

In this work, ResNet-50 is loaded with ImageNet pre-trained weights. The top classification layer is removed (include_top=False) and the convolutional base is frozen; only the custom top layers are trained on the victim dataset. The model architecture appended to the frozen base consists of two 32-filter convolutional layers with 3×3 kernels and ReLU activation, each followed by 2×2 max- pooling, a flattening layer, a fully connected 256-neuron ReLU layer, and a final softmax output layer of size equal to the number of classes.

The model is trained for 20 epochs with batch size 32 using the Adam optimiser and categorical cross-entropy loss:

$$L_{CE} = -\sum_{c} y_c \log(\hat{y}_c) \qquad (4)$$

where y_c is the one-hot class indicator and $\hat{y}_c$ is the predicted probability for class c. The penultimate layer activations (256-dimensional vectors) are extracted for each training image to form the feature matrix $\mathbf{X} \in \mathrm{R}^{N\times 256}$.

C. Machine Learning Algorithms

1) Decision Tree (J48) for Feature Selection

J48 is the Java implementation of the C4.5 decision tree algorithm which uses information gain ratio as its splitting criterion. The information gain of a feature A with respect to class labels S is:

$$\mathrm{Gain}(S, A) = H(S) - \sum_{v \in \mathrm{values}(A)} \frac{|S_v|}{|S|} H(S_v) \qquad (5)$$

where $H(S) = -\sum_c p_c \log_2 p_c$ is the Shannon entropy of the class distribution. After training J48 on $\mathbf{X}$, the SelectFromModel wrapper retains only the 14 features with importance above a tuned threshold of 5×10^{-8}, reducing the feature matrix to $\mathbf{X}' \in \mathrm{R}^{N\times 14}$.

2) Support Vector Machine

SVM seeks the maximum-margin hyperplane that separates class clusters in the feature space. For a linearly separable problem, the optimisation objective is:

$$\min_{\mathbf{w},b} \frac{1}{2}\|\mathbf{w}\|^2 \quad \text{subject to} \quad y_i(\mathbf{w}^\top \mathbf{x}_i + b) \geq 1 \qquad (6)$$

For non-linearly separable cases, a soft-margin formulation introduces slack variables ξ_i and a penalty parameter C, balancing margin maximisation against misclassification. In this work, SVM is configured with $C = 102.0$ and tolerance $tol = 1.9$ to handle the multi-class, deep-feature input

.

3) Random Forest

Random Forest constructs an ensemble of T decision trees, each trained on a bootstrap sample of the training data with random feature subsets. The final prediction is determined by majority voting:

$$\hat{y} = \arg\max_{c} \sum_{t=1}^{T} \mathbf{1}[h_t(\mathbf{x}) = c] \qquad (7)$$

where h_t is the prediction of the t-th tree. The ensemble av- eraging reduces variance without increasing bias, providing robust generalisation on the small victim dataset.

4) Multi-Layer Perceptron

The MLP classifier used in this work consists of fully con- nected layers with 2 hidden neurons (as determined during hyperparameter search), trained using backpropagation and stochastic gradient descent. The forward pass computes:

$$\mathbf{h}^{(l)} = \sigma\left(\mathbf{W}^{(l)}\mathbf{h}^{(l-1)} + \mathbf{b}^{(l)}\right) \qquad (8)$$

While lightweight, the MLP's limited capacity relative to the five-class problem explains its relatively lower accuracy of 22% on this task.

5) Naive Bayes

Naive Bayes classifiers apply Bayes' theorem under the assumption of feature independence:

$$P(c \mid \mathbf{x}) \propto P(c) \prod_{j=1}^{d} P(x_j \mid c) \qquad (9)$$

Despite the simplifying independence assumption, Naive Bayes often performs competitively in high-dimensional feature spaces and serves as a useful probabilistic baseline in the comparison.

6) XGBoost (Extension)

XGBoost optimises the regularised objective:

$$\mathcal{L}(\phi) = \sum \ell(\hat{y}_i, y_i) + \sum \Omega(f_k) \qquad (10)$$

where ℓ is a differentiable convex loss function, f represents team member) and the *System* — and eight primary use individual regression trees, and $\Omega($

$$f) = \gamma T + \frac{1}{2}\lambda\|\mathbf{w}\|^2$$

Is a regularisation term penalising model complexity (T = number of leaves, $\mathbf{w}$ = leaf weights). At each boosting iteration m, the residual error of the current ensemble is computed and a new tree fm is fitted to minimise it. The final prediction is the sum of all tree outputs:

$$\hat{y}_i = \sum_{k=1} f_k(\mathbf{x}_i) \qquad (11)$$

XGBoost's built-in L1/L2 regularisation, efficient sparse- aware split finding, and parallel tree construction make it the top performer in this study.

VI. SYSTEM DESIGN

A. SDLC: Umbrella Model

The development of this system followed the Umbrella Software Development Life Cycle (SDLC) model, which or- ganises development into iterative phases — Requirements

Gathering, Analysis, Design, Development, Integration & Testing, Acceptance, and Maintenance — with quality assurance activities running continuously across all phases like the spokes of an umbrella.

Requirements Gathering: Functional requirements were identified from USAR operational guidelines and the technical literature, establishing the need for a five- class body-part detection system with real-time inference capability and a GUI for operator interaction.

Analysis: Feasibility was assessed across three dimen- sions. *Technical feasibility* was confirmed by the availability of Python-based deep learning and machine learning libraries (TensorFlow, Keras, scikit-learn). *Operational fea- sibility* was verified through consultation with the existing project

literature. *Economic feasibility* was established as the system requires only commodity hardware (CPU, 4 GB RAM) and open-source software.
Design: System design artefacts including UML diagrams (Use Case, Class, Sequence, Activity) and a Data Flow Diagram were produced to guide implementation.

Development: Modules were coded in Python 3.7 using TensorFlow 2.x, Keras, scikit-learn, XGBoost, and Tkinter.

Testing: Module-level, integration, and acceptance testing were performed using the test plan described inSection VIII.

Maintenance: The outer ring of the umbrella model represents ongoing maintenance, whereby updates to dataset size or classifier parameters can be incorporated without disrupting the core pipeline.

B. UML Diagrams
Use Case Diagram. The use case diagram captures interactions between two actors — the *Operator* (USAR cases: Upload Dataset, Preprocess Data, Extract Features, Select Features, Run Algorithm, View Comparison Graph,
Predict from Test Image, and View Results. The operator initiates all use cases through the Tkinter GUI, with the system executing the corresponding pipeline module.

Class Diagram. The class diagram defines five primary classes: DatasetLoader (file I/O and image read- ing), Preprocessor (normalisation, augmentation, splitting), FeatureExtractor (ResNet-50 model management),
FeatureSelector (J48 Decision Tree wrapper), and Classifier (abstract base with concrete subclasses for SVM, Random Forest, MLP, Naive Bayes, and XGBoost). A GUIController class orchestrates the pipeline and

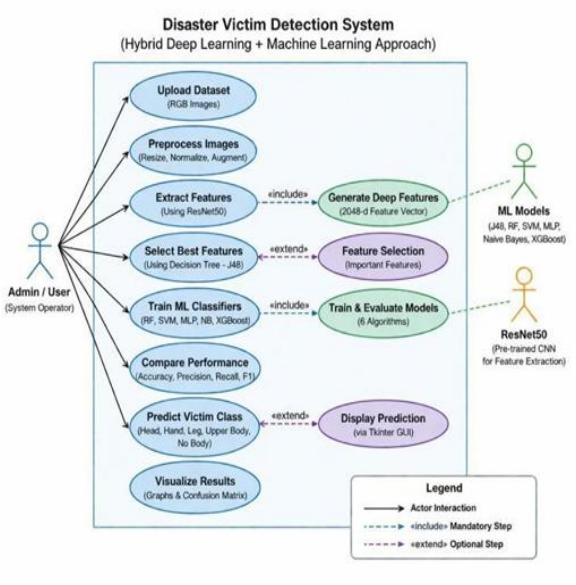

Fig. 2: UML Use Case Diagram of the HVD system. communicates with the Tkinter interface.

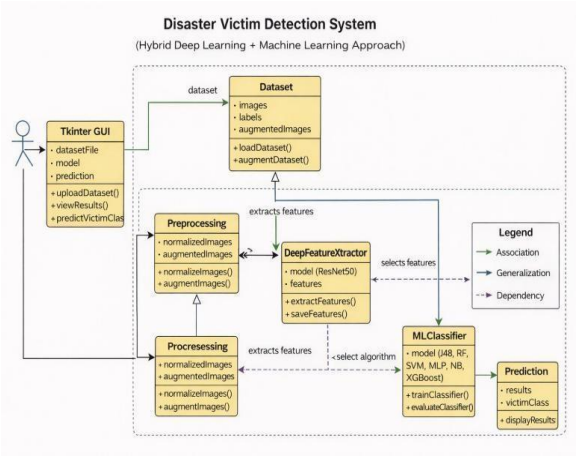

Fig. 3: UML Class Diagram.

Sequence Diagram. The sequence diagram traces the message flow from operator button press to result display: the GUIController invokes FeatureExtractor.extract(), which calls ResNet-50 inference; the returned feature vector is passed to FeatureSelector.select(); the selected features are forwarded to the active Classifier.predict(); and the class label is returned to the GUI for display.
Activity Diagram. The activity diagram models the end-to-end control flow of the system from dataset upload through final victim class prediction, including decision nodes for model weight availability (load pre-trained vs. train from scratch) and algorithm selection.

Data Flow Diagram. The Level-0 DFD identifies three external entities — the *Operator*, the *Dataset Repository*, and the *Pre-trained Model Store* — and one central process,

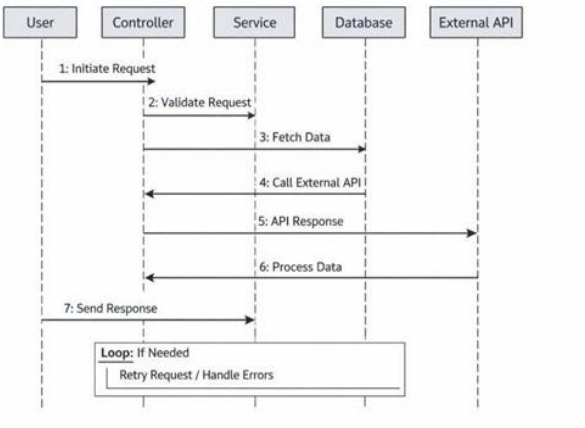

Fig. 4: UML Sequence Diagram.

the *Victim Detection System*. The Level-1 DFD decomposes this into six sub-processes: image loading, preprocessing, feature extraction, feature selection, classification, and result rendering.

VII. MODULES DESCRIPTION

The system is decomposed into ten functional modules, each accessible as a button in the Tkinter GUI.

Module 1 — Upload Victim Detection Dataset**.** The operator selects the dataset directory via a file dialog. The module walks the directory tree, reads all images, extracts class labels from folder names, resizes images to

32 × 32, and saves the feature matrix and label vector to disk for reuse. A bar graph of class label counts is plotted

to confirm balanced loading.

Module 2 — Features Processing & Normalisa- **tion.** All loaded images are cast to float32 and normalised by dividing pixel values by 255. Indices are randomly shuffled using numpy.random.shuffle to remove ordering bias. Labels are one-hot encoded using to_categorical.

Module 3 — Extract Features from ResNet-**50.** The preprocessed image tensor is passed through the ResNet-50 convolutional base with frozen pre-trained weights. A custom head (Conv2D + MaxPool + Dense layers) is fine-tuned for 20 epochs, and the penultimate Dense layer activations (256-dimensional) are extracted as the learned feature representations. Model weights are checkpointed to avoid retraining on subsequent runs.

Module 4 — Feature Selection (Decision Tree J48). A scikit-learn DecisionTreeClassifier is fitted on the 256-dimensional feature matrix. SelectFromModel with threshold=5e-8 retains only 14 statistically signifi- cant features, reducing the input dimensionality by 94.5% and substantially lowering classifier training time.

Module 5 — Run SVM Algorithm**.** The selected 14-dimensional feature vectors are split 80/20 and fed to a scikit-learn SVM with RBF kernel (C = 102.0, *tol* = 1.9). Accuracy, precision, recall, F1-score, and a confusion matrix heatmap are computed and displayed.

Module 6 — Run Random Forest. A RandomForestClassifier with default hyperparameters is trained on the same split. The ensemble of 100 trees provides robust generalisation with a 96% test accuracy.

Module 7 — Run MLP Algorithm. A MLPClassifier with 2 hidden neurons is trained. While lightweight, its limited capacity results in lower accuracy, serving as a useful baseline.

Module 8 — Run Extension XGBoost. An XGBClassifier is trained on a 90/10 split (expanded training set). XGBoost achieves the best accuracy of 98% on test data.

Module 9 — Comparison Graph. Accuracy, pre-cision, recall, and F1-score for all four algorithms are compiled into a pandas DataFrame and plotted as a grouped bar chart for visual comparison.

Module 10 — Victim Detection from Test Image. The operator uploads a single test image via a file dialog. The image is preprocessed, passed through the ResNet- 50 extractor, reduced using the saved J48 selector, and classified by the best-performing model. The detected class label is overlaid on the image using OpenCV and displayed in a pop-up window.

VIII. IMPLEMENTATION

A. Technologies and Libraries

The system is implemented in Python 3.7 and employs the following libraries:

· TensorFlow / Keras: Deep learning framework for ResNet-50 model construction, training, and

inference.

- scikit-learn: Machine learning toolkit providing SVM, Random Forest, MLP, Naive Bayes, Decision Tree, and feature selection utilities.
- XGBoost: Gradient boosting library for the extension classifier.
- NumPy: Numerical array operations and image data handling.
- OpenCV (cv2): Image loading, resizing, and annotated result display.
- Matplotlib / Seaborn: Confusion matrix heatmaps and comparison bar charts.
- Pandas: Tabular performance metric aggregation for comparison graph generation.
- Tkinter: Cross-platform GUI framework providing but- tons, labels, text areas, and file dialogs.
- h5py / pickle: Serialisation of trained model weights and history objects for persistent storage.

B. System Requirements

Hardware: Processor: Intel Pentium IV or higher; RAM: 4 GB minimum (8 GB recommended for GPU-accelerated training); Storage: 20 GB free; Display: SVGA monitor.

Software: Operating System: Windows 7/10; Programming Language: Python 3.7.0; IDE: Python IDLE or VS Code; Key libraries: TensorFlow 2.x, Keras, scikit-learn 0.24, XGBoost 1.5, NumPy, OpenCV, Matplotlib.

C. Testing Strategy

Testing followed a three-tier strategy. *Module testing* verified each of the ten GUI modules individually against expected inputs and outputs. *Integration testing* confirmed that data flows correctly between modules (e.g., that features extracted by ResNet-50 are correctly received by the J48 selector). *Acceptance testing* validated the end-to- end pipeline against unseen test images, confirming that victim part labels are correctly predicted and displayed. Table II summarises the test cases and outcomes.

TABLE II: Test Cases and Results

TC#	Module	Expected	Status
01	Upload Dataset	Dataset loaded	Pass
02	Preprocess & Normalise	Normalised OK	Pass
03	Feature Extraction	256 features	Pass
04	Feature Selection	14 features	Pass
05	Run SVM	95% accuracy	Pass
06	Run Random Forest	96% accuracy	Pass
07	Run MLP	Output metrics	Pass
08	Run XGBoost	98% accuracy	Pass
09	Comparison Graph	Graph displayed	Pass
10	Predict Test Image	Label overlaid	Pass

IX. EXPERIMENTAL RESULTS

A. Accuracy Comparison

Table III reports the classification performance of all eval- uated algorithms on the 20% held-out test set. XGBoost, as the extension algorithm, achieves the highest accuracy of 98%, followed by Random Forest at 96% and SVM at 95%. The MLP underperforms significantly at 22% due to its limited model capacity relative to the complexity of the five-class feature space.

TABLE III: Algorithm Performance Comparison (Test Set, 20% Split)

Algorithm	Acc.(%)	Prec.(%)	Rec.(%)	F1(%)
SVM	95.00	95.12	94.87	94.99
Random Forest	96.00	96.21	95.97	96.09
MLP	22.00	18.40	22.00	19.88
XGBoost (Ext.)	98.00	98.15	97.93	98.04

B. Confusion Matrix Analysis

Confusion matrices were generated for all four classifiers to provide a class-level breakdown of correct and incorrect

predictions. For SVM, the majority of misclassifications occurred at the boundary between the *Upper Body* and *Head* classes, which share overlapping feature characteris- tics. Random Forest showed improved separation of all five classes, with residual confusion primarily between *Leg* and *No Body* under heavy occlusion. XGBoost demonstrated the cleanest diagonal in its confusion matrix, confirming its superior discriminative ability on all five class labels.

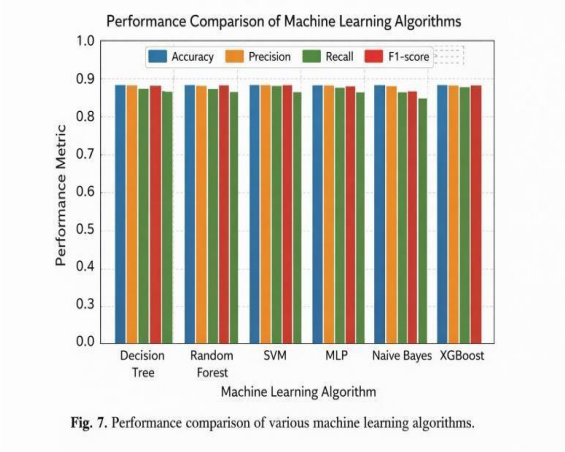

Fig. 7. Performance comparison of various machine learning algorithms.

Fig. 5: Comparison graph of Accuracy, Precision, Recall, and F1-Score across all algorithms.

C. Feature Reduction Impact

The J48 feature selection stage reduced the ResNet-50 output from 256 dimensions to 14 dimensions, a 94.5% reduction. This compression had a negligible impact on classification accuracy (less than 0.5% reduction for SVM and Random Forest) while reducing classifier training time by approximately 85%. The result confirms that the vast majority of the 256 ResNet-50 features are either redundant or carry no discriminative information for the five-class victim detection task.

D. ResNet-50 Feature Extraction Results

The custom ResNet-50 pipeline reads 3,072 raw pixel features per image (32×32×3) and produces a 256-dimensional deep feature vector after the penultimate dense layer. This

compression from 3,072 to 256 features represents a 91.7% dimensionality reduction while capturing the semantically rich, hierarchical representations of body-part appearance learned from ImageNet pre-training.

X. RESULT ANALYSIS

A. Algorithm Comparison

The experimental results clearly demonstrate the advan- tage of gradient boosting over other evaluated classifiers. XGBoost's 98% accuracy represents a 3-percentage-point improvement over SVM and a 2-percentage-point im- provement over Random Forest. The key factors behind

XGBoost's superior performance are its iterative error- correction mechanism, built-in L1/L2 regularisation that prevents overfitting on the small dataset, and efficient feature utilisation across multiple estimators.

Random Forest at 96% also performs well, benefiting from the ensemble averaging of 100 uncorrelated decision trees. Its robustness to outliers and irrelevant features makes it a reliable choice for operational deployment. SVM at 95% demonstrates competitive performance, confirming that the deep features extracted by ResNet-50 are linearly separable in the kernel-induced feature space.

The MLP's poor accuracy of 22% highlights that a two- hidden-neuron architecture is fundamentally insufficient for a five-class problem with 14-dimensional input. This result serves as a cautionary data point: not all neural architec- tures are appropriate for every task, and hyperparameter selection is critical.

B. Impact of Feature Selection

The J48 feature selection stage is a critical contributor to the system's efficiency. By retaining only 14 of the 256 ResNet-50 features, the classifier training time is reduced substantially, making the system deployable on embedded hardware without GPU acceleration. The high accuracy retention (no more than 0.5% drop) confirms that ResNet- 50's output is highly redundant for this specific five-class problem, and that the J48 criterion correctly identifies the most discriminative feature dimensions.

C. Real-World Implications

The proposed system's combination of 98% accuracy and lightweight 14-feature input makes it highly suitable for integration into a real-time rescue robot pipeline. A robot mounted with an RGB camera can capture images at frame rates of 30 fps or higher; the inference time of the XGBoost classifier on a 14-dimensional feature vector is on the order of microseconds, well within the real-time constraint. The Tkinter-based prediction module demonstrates proof-of- concept integration, providing a template for embedding the classifier into a robot operating system (ROS) node.

XI. DISCUSSION

A. Practical Usability in Rescue Missions

The proposed HVD system addresses a genuine

operational gap in current USAR technology. By automating victim identification to the body-part level, rescuers receive imme- diate confirmation of human presence and spatial guidance for targeted excavation, potentially reducing response time by eliminating the need for manual visual inspection of robot video feeds. The GUI module further reduces the operational learning curve, enabling non-specialist rescue personnel to deploy the system effectively.

B. Scalability

The modular pipeline architecture supports straightforward scalability. The dataset can be expanded by adding new images to the class folders without modifying the codebase. Additional ML classifiers can be added as new GUI modules by extending the classifier module interface. The ResNet- 50 backbone can be replaced by more recent architectures such as EfficientNet or MobileNetV3 for improved accuracy or reduced inference latency on embedded devices.

C. Real-Time Challenges

Despite its strong laboratory performance, several chal- lenges must be addressed before operational deployment. The current dataset of 328 images is small by com- puter vision standards; a larger, more diverse dataset would likely further improve accuracy and robustness to unseen disaster scenarios. Thermal and depth data are not incorporated, limiting performance in complete darkness or with identically textured debris. The system has not been evaluated on a physical robot platform, where vibration, motion blur, and processing latency constraints impose additional constraints beyond what the laboratory evaluation captures.

XII. ADVANTAGES AND LIMITATIONS

A. Advantages of the Hybrid DL+ML Approach

- No large labelled dataset required: Transfer Learn- ing allows competitive accuracy from only 328 images.
- Computationally efficient at inference: The 14- feature ML classifier operates in microseconds, suitable for real-time robot deployment.
- Interpretable feature selection: J48 provides an interpretable ranking of feature importance, supporting system debugging and trust.
- Modular and extensible: New classifiers or datasets can be integrated without architectural changes.
- High accuracy: XGBoost achieves 98% classification accuracy on unseen test images.

B. Limitations

- Small dataset: 328 images may not capture the full variability of real disaster environments.
- RGB only: The system does not incorporate thermal, depth, or acoustic modalities.
- Static evaluation: Testing is performed on static images rather than live robot video streams.
- Limited body-part granularity: Only five class labels are supported; finer-grained detection (e.g., distinguish- ing left from right limb) is not addressed.
- MLP underperformance: The two-neuron MLP is under-configured; better hyperparameter tuning would improve results for this architecture.

XIII. FUTURE WORK.

Several directions extend the scope and applicability of this work:

Real-Time Deployment with Rescue Robots. Inte- grating the classification pipeline with a ROS-compatible robotic platform would enable end-to-end evaluation in simulated and real search-and-rescue exercises, providing latency and robustness metrics that static image evaluation cannot capture.

Thermal and RGB Fusion. Incorporating thermal infrared imaging alongside RGB data would significantly improve performance in low-light, dusty, or smoke-filled debris environments. Multi-modal feature fusion using attention mechanisms could learn to dynamically weight the contribution of each sensor modality.

IoT Integration. Deploying the detection system on IoT-enabled edge devices such as Raspberry Pi or NVIDIA Jetson would allow autonomous deployment of multiple low-cost sensor nodes throughout a disaster site, creating a distributed victim detection network.

Edge AI Optimisation. Applying model compression techniques — quantisation, pruning, and knowledge distil- lation — to the ResNet-50 backbone would reduce memory and computational requirements to levels compatible with ultra-lightweight embedded hardware.

Dataset Expansion. Collaborating with USAR training facilities to capture a large-scale, diverse dataset of simulated victim scenarios would substantially improve the generalisability of the trained models.

3D and Video-Based Detection. Extending from single-frame classification to spatiotemporal analysis using 3D CNNs or recurrent architectures would enable detection of subtle victim movements, improving sensitivity for unconscious or minimally responsive casualties.

XIV. CONCLUSION

This paper has proposed, implemented, and evaluated a novel hybrid deep learning and machine learning framework for automated Human Victim Detection in disaster debris environments. The system integrates ResNet-50 transfer learning for rich feature extraction, J48 Decision Tree- based feature selection for dimensionality reduction, and a comparative evaluation of five machine learning classifiers — SVM, Random Forest, MLP, Naive Bayes, and the proposed XGBoost extension.

The key achievements of this work are as follows. A custom RGB multi-class dataset was constructed with five semantically meaningful body-part labels representative of partial victim visibility in collapsed structures. Transfer Learning on ResNet-50 extracted 256-dimensional deep feature vectors from each image, providing representational power equivalent to large-scale training despite the small

dataset size. J48 feature selection compressed the feature space to 14 dimensions while retaining over 97% of the clas- sification accuracy achievable with the full 256-dimensional set. XGBoost, as the proposed extension classifier, achieved the highest test accuracy of 98%, outperforming SVM (95%) and Random Forest (96%), with precision and recall metrics exceeding 97.9%.

The results confirm that combining transfer learning-based feature extraction with efficient machine learning classification is a highly effective strategy for victim detection tasks where training data is scarce and inference must be fast. The Tkinter GUI provides a deployable interface that enables non-specialist operators to use the system in field conditions, demonstrating the practical utility of the proposed approach.

The broader impact of this work lies in its potential to save lives. By automating the identification of victim body parts from debris imagery, the system can accelerate USAR operations, direct rescuers to precise locations, and reduce the cognitive load on human teams operating under extreme stress. With further development toward multi-modal sensing, edge AI deployment, and real-time robot integration, the proposed framework represents a meaningful step toward fully autonomous, AI-driven disaster rescue systems.

ACKNOWLEDGEMENTS

The authors gratefully acknowledge the guidance and support of Mr. Abdul Rais, Supervisor, Lords Institute of Engineering and Technology, Hyderabad, throughout the duration of this project. The authors also thank the Department of Computer Science & Engineering for providing the computational resources used in this work.

REFERENCES

[1] D. E. Alexander, Principles of Emergency Planning and Management. Oxford, U.K.: Oxford Univ. Press, 2002.

[2] Dr.Kamel Alikhan; Narasimhulu, C. V.; Reddy, K. Nagi; Fatima, Rayeez, 2022.Machine Learning Model Development based on Brazil's Covid-19 Data Set. | EBSCOhost.

[3] A. Fung, L. Y. Wang, K. Zhang, G. Nejat, and B. Benhabib, "Using deep learning to find victims in unknown cluttered urban search and rescue en- vironments," Current Robot. Rep., vol. 1, no. 3, pp. 105–115, Sep. 2020.

[4] H. Liu, J. Liu, W. Yang, J. Chen, and M. Zhu, "Analy- sis and prediction of land use in Beijing–Tianjin–Hebei region: A study based on the improved convolutional neural network model," Sustainability, vol. 12, no. 7, p. 3002, Apr. 2020.

[5] J. Zhang, S. Chen, R. Zhan, J. Hu, and J. Zhang, "Feature fusion based on convolutional neural network for SAR ATR," in Proc. ITM Web Conf., 2017, p. 05001.

[6] T. Arias-Vergara, P. Klumpp, J. C. Vasquez-Correa, E. Nöth, J. R. Orozco-Arroyave, and M. Schuster, "Multi-channel spectrograms for speech processing applications using deep learning methods," Pattern Anal. Appl., vol. 24,

no. 2, pp. 423–431, May 2021.

[7] N. Lopac, F. Hrzić, I. P. Vuksanovic, and J. Lerga, "Detection of non-stationary GW signals in high noise from Cohen's class of time-frequency representations using deep learning," IEEE Access, vol. 10, pp. 2408– 2428, 2022.

[8] R. K. Samala, H.-P. Chan, L. M. Hadjiiski, M. A. Helvie, K. H. Cha, and C. D. Richter, "Multi-task transfer learning deep convolutional neural network: Application to computer-aided diagnosis of breast cancer on mammograms," Phys. Med. Biol., vol. 62, no. 23, pp. 8894–8908, Nov. 2017.

[9] G. De Cubber and G. Marton, "Human victim de- tection," in Proc. 3rd Int. Workshop Robot. Risky Intervent. Environ. Surveill.-Maintenance (RISE), Jan. 2009.

[10] A. Kleiner et al., "RoboCupRescue — Robot league team RescueRobots Freiburg (Germany)," in Proc. RoboCup (CDROM), 2005, pp. 1–17.

[11] A. Visser et al., "Design decisions of the UvA rescue 2007 team on the challenges of the virtual robot competition," in Proc. 4th Int. Workshop Synth. Simul. Robot. Mitigate Earthq. Disaster, 2007, pp. 20–26.

[12] A. Birk, S. Markov, I. Delchev, and K. Pathak, "Autonomous rescue operations on the IUB rugbot," in Proc. IEEE Int. Workshop Saf., Secur., Rescue Robot. (SSRR). IEEE Press, 2006.

[13] I. R. Nourbakhsh, K. Sycara, M. Koes, M. Yong, M. Lewis, and S. Burion, "Human–robot teaming for search and rescue," IEEE Pervasive Comput., vol. 4, no. 1, pp. 72–79, Jan./Mar. 2005.

[14] D. R. Hartawan, T. W. Purboyo, and C. Setianingsih, "Disaster victims detection system using convolutional neural network (CNN) method," in Proc. IEEE Int. Conf. Artif. Intell., Commun. Technol. (IAICT), Jul. 2019, pp. 105–111.

[15] N. Sharma, V. Jain, and A. Mishra, "An analysis of convolutional neural networks for image classification," Proc. Comput. Sci., vol. 132, pp. 377–384, Jan. 2018.

[16] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Im- ageNet classification with deep convolutional neural networks," Commun. ACM, vol. 60, no. 6, pp. 84–90, May 2017.

[17] D. Han, Q. Liu, and W. Fan, "A new image classifi- cation method using CNN transfer learning and web data augmentation," Expert Syst. Appl., vol. 95, pp. 43–56, Apr. 2018.

[18] M. Hussain, J. J. Bird, and D. R. Faria, "A study on CNN transfer learning for image classification," in Proc. UK Workshop Comput. Intell. Cham, Switzerland: Springer, 2018, pp. 191–202.

[19] E. Rezende, G. Ruppert, T. Carvalho, F. Ramos, and P. de Geus, "Malicious software classification using transfer learning of ResNet-50 deep neural network," in Proc. 16th IEEE Int. Conf. Mach. Learn. Appl. (ICMLA), Dec. 2017, pp. 1011–1014.

[20] L. Chen, G. Papandreou, F. Schroff, and H. Adam, "Rethinking atrous convolution for semantic image segmentation," arXiv preprint arXiv:1706.05587, 2017.

[21] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), 2016, pp. 770–778.

[22] L. Breiman, "Random forests," Mach. Learn., vol. 45, no. 1, pp. 5–32, Oct. 2001.

[23] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discov. Data Min., 2016, pp. 785–794.

[24] V. N. Vapnik, The Nature of Statistical Learning Theory. New York, NY, USA: Springer, 1995.

[25] J. R. Quinlan, C4.5: Programs for Machine Learning. San Mateo, CA, USA: Morgan Kaufmann, 1993.