International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 09 Issue: 12 | Dec - 2025 SJIF Rating: 8.586 ISSN: 2582-3930

Smart Health Analysis System Using Machine Learning and Deep Learning (Ayusense)

G V Sowmya[1], Bhoomika MD[2], Brunda V[2], Chaithra R[2], and Dhanya R[2]

[1]Assistant Prof., Dept. of ISE, JNNCE, Shivamogga, India.

[2]UG students, Dept. of ISE, JNNCE, Shivamogga, India.

Abstract - AyuSense is a Smart Health Analysis System designed to automate the detection and monitoring of various medical conditions using advanced machine learning and deep learning techniques. The system integrates multiple types of health data, including clinical tabular data (such as Complete Blood Count and other laboratory results), medical images (chest X-rays, CT scans, and skin lesion images), and biosignals like 12-lead ECG recordings. Machine learning algorithms, such as Random Forests, are employed for structured clinical data analysis, while deep learning architectures, including Convolutional Neural Networks (CNNs) and pretrained models like MobileNetV2 and ResNet50, are utilized for medical image classification and ECG signal analysis. The system includes preprocessing, normalization, data augmentation, and multi-class/multi-label prediction pipelines to enhance model performance and reliability. Extensive evaluation across multiple datasets demonstrates high predictive accuracy, F1-scores, and area-under-curve (AUC) metrics, validating the system's capability to provide accurate and automated health assessments. By combining diverse medical data modalities into a unified framework, AyuSense offers a scalable and interpretable platform for comprehensive health analysis and automated diagnostic prediction.

Key Words : Smart Health Analysis, Machine Learning, Deep Learning, Multi-modal Medical Data, Medical Image Classification, ECG Signal Analysis, Automated Diagnostic Prediction.

1.INTRODUCTION

The healthcare sector is increasingly relying on technology to improve diagnostic accuracy, patient monitoring, and overall treatment efficiency. Traditional methods of health assessment, which depend heavily on manual evaluation of clinical reports, medical images, and biosignals, are often time-consuming and prone to human error. To address these limitations, the integration of artificial intelligence (AI) techniques such as machine learning (ML) and deep learning (DL) has emerged as a promising solution for automated health analysis.

AyuSense is a Smart Health Analysis System designed to provide a comprehensive, automated platform for analyzing diverse medical data and predicting potential health conditions. The

system consolidates three major types of medical data: clinical tabular data (including Complete Blood Count and other laboratory results), medical images (chest X-rays, CT scans, and skin lesion images), and biosignals (12-lead ECG recordings). By integrating these multi-modal data sources, AyuSense is capable of performing accurate, multi-class, and multi-label health predictions, enabling a broader understanding of patient health conditions.

For structured clinical data, machine learning models such as Random Forest are employed to identify patterns and anomalies in laboratory reports. For medical image analysis, deep learning architectures including Convolutional Neural Networks (CNNs) and pretrained models like MobileNetV2 and ResNet50 are used to extract high-level features and classify images effectively. ECG signal analysis is also performed using deep learning models to detect various cardiac conditions from raw biosignals. The system applies rigorous preprocessing, normalization, and data augmentation techniques to enhance model performance, prevent overfitting, and ensure robustness across different datasets.

AyuSense emphasizes not only accuracy but also scalability and interpretability. Multi-label prediction pipelines allow the system to handle complex scenarios where multiple health conditions may coexist, while evaluation metrics such as accuracy, F1-score, and area-under-curve (AUC) validate its performance across diverse datasets. By automating the analysis of heterogeneous medical data in a single framework, AyuSense reduces manual effort, accelerates diagnostic processes, and enables proactive health monitoring.

The system has potential applications in personalized healthcare, remote patient monitoring, and early disease detection, offering a foundation for data-driven decision-making. By providing an integrated platform for real-time health analysis, AyuSense aims to enhance patient outcomes and support the development of intelligent healthcare solutions for modern medical practices.

2. RELATED WORK

In recent years, automated health analysis systems have emerged as a promising approach for early disease detection, continuous monitoring, and decision support in healthcare. These systems leverage machine learning, deep learning, and computer vision techniques to analyze diverse medical data, including images, signals, and clinical records. Several studies have explored these approaches across multiple domains, highlighting their effectiveness and potential for integration into unified health platforms.

Deep learning research in healthcare has expanded rapidly, with multiple studies focusing on dermatological imaging, cardiovascular signals, radiology, hematology, and smart healthcare applications. The following section summarizes the extracted insights from twelve

recent studies and highlights their relevance to the development of AyuSense, a unified smart health analysis platform.

Aquil et al. [1] introduced a hybrid machine learning and deep learning system aimed at early skin disease detection across diverse skin tones. Their work emphasized dataset diversity, addressing the long-standing issue of bias in dermatology datasets. The study demonstrated that combining handcrafted features with CNN embeddings significantly improves sensitivity for darker skin tones, an insight directly relevant to AyuSense's skin disease module, which aims for inclusiveness and real-world clinical usability.

Chen et al. [2] proposed AI-Skin, a closed-loop self-learning framework that automatically expands its dataset through user feedback and continual learning. Their approach enables the model to improve over time without manual retraining. This aligns with AyuSense's long-term goal of incremental model enhancement through user-uploaded data, ensuring continuous performance improvement.

Noor et al. [3] improved skin disease classification using dataset refinement and attention-based vision models. Their findings highlight that cleaning mislabeled samples and applying attention mechanisms such as CBAM/SE blocks can boost classification accuracy in multi-class dermatology tasks. AyuSense incorporates this insight by applying data augmentation, attention models, and refined preprocessing strategies to improve lesion identification.

In cardiovascular analysis, Makhmudov et al. [4] developed a multitask deep learning model to predict complications following myocardial infarction. Their model jointly learns multiple risk factors, demonstrating that multitask networks provide deeper clinical insights and reduce computational cost. This supports AyuSense's ECG module, which integrates rhythm classification and abnormality detection through CNN-LSTM architectures.

Golande and Pavankumar [5] introduced an optical ECG–based deep learning model combining signal feature extraction and CNN classification. Their hybrid approach achieved higher accuracy for heart disease prediction and highlighted the importance of noise filtering and feature fusion in ECG analysis. These extracted techniques inform AyuSense's ECG pipeline, particularly in preprocessing (noise removal) and hybrid model design.

For respiratory imaging, Sharma and Guleria [6] used VGG-16 combined with a dense neural classifier for detecting pneumonia from chest X-rays. Their model achieved high sensitivity due to the depth of VGG-16's convolutional layers, proving that transfer learning works exceptionally well for medical imaging tasks with limited datasets. Bharati et al. [7] expanded on this by designing a hybrid deep learning

classifier capable of diagnosing various lung diseases, demonstrating that combining multiple CNN models enhances feature extraction. These findings shape AyuSense's radiology module, which uses CNN and Grad-CAM for pneumonia, TB, and lung abnormality detection.

In the domain of kidney imaging, Zhang et al. [8] reviewed advancements in deep learning approaches for nephrology, emphasizing segmentation, tumor detection, and functional renal assessment using CNN and transformer-based architectures. Their work identifies challenges such as limited annotated data and the need for clinically interpretable AI outputs. AyuSense addresses these concerns by integrating explainability through Grad-CAM heatmaps and modular training for CT scan classification.

Reghunandanan et al. [9] proposed a CNN model capable of detecting reticulocytes in peripheral blood smears. Their study demonstrated AI's capability to automate microscopic analysis with high precision, significantly reducing manual workload. This is relevant to AyuSense's blood report module, which uses ML/DL algorithms (XGBoost, LightGBM) to classify hematological abnormalities.

In neuroimaging, Chattopadhyay and Maitra [10] developed a CNN-based model for MRI-based brain tumor detection. Their model achieved strong classification accuracy and demonstrated that deep learning can outperform traditional radiological methods, especially in complex tumor segmentation tasks. This contributes to AyuSense's CT/MRI module design choices, particularly CNN-based feature extraction and classification.

Beyond imaging and diagnostics, Lv et al. [11] introduced a deep learning-based predictive evaluation framework for smart healthcare systems, using multimedia input for patient monitoring. Their model demonstrated that multimodal sensing and AI integration significantly improve healthcare responsiveness. AyuSense adopts this concept by unifying multiple datasets (ECG, images, blood parameters) into one intelligent platform.

Finally, Chen [12] explored deep learning–assisted user interface design for senior healthcare applications. Their study highlighted the importance of usability, accessibility, and age-appropriate interface structures for health app adoption. These findings support AyuSense's goal to provide a clean, user-friendly, and inclusive dashboard suitable for both general users and clinical professionals.

Collectively, the extracted findings from these studies indicate strong advancements in disease detection, multimodal learning, and healthcare automation. However, most existing research focuses on single-dataset or single-disease prediction. A gap exists in unified, multi-model healthcare platforms capable of handling images,

signals, and structured medical data concurrently. AyuSense addresses this gap by integrating dermatology, cardiology, radiology, hematology, and CT imaging into a single AI-enabled smart diagnostic system.

3.METHODOLOGY

The Smart Health Analysis System (AyuSense) is designed to provide real-time health analysis using multiple types of medical data. The system is implemented in four main phases: Data Collection, Model Training, Prediction, and Dashboard Integration.

Data Collection:

Data is gathered from multiple publicly available datasets to ensure diversity and robustness. This includes the Complete Blood Count (CBC) dataset for blood test parameters, skin disease image datasets for dermatological conditions, chest X-ray datasets for Pneumonia and Covid-19, and the PTB-XL ECG dataset for cardiac signals. All collected data are preprocessed according to type: numerical blood parameters are cleaned and normalized, images are resized and augmented, and ECG signals are padded and normalized to a fixed length.

Model Training:

Different algorithms are applied based on the dataset type. CBC blood data is scaled using StandardScaler, and disease labels are encoded using LabelEncoder, followed by training a Random Forest Classifier for disease classification. Skin disease and chest X-ray images are processed using Convolutional Neural Networks (CNNs) for feature extraction and classification. ECG signals are processed using 1D CNNs with Multi-Label encoding to identify multiple cardiac conditions in a single record. Models are trained with appropriate optimizers and loss functions (e.g., binary cross-entropy for ECG multi-label classification), and performance is evaluated using metrics such as accuracy, classification reports, F1-scores, and confusion matrices. Data augmentation and regularization are applied as necessary to improve model generalization.

Prediction:

The system allows single-report, real-time prediction via a Tkinter interface. CBC reports in PDF or image format are processed using pdfplumber and pytesseract, and relevant blood parameters are extracted using regular expressions before being passed to the trained Random Forest model. Skin and chest X-ray images, as well as ECG signals, are preprocessed and classified using their respective trained CNN or Multi-Label CNN models. The Tkinter interface displays the predicted disease, probability scores, and feature highlights immediately after processing the uploaded report.

Dashboard Integration:

A Streamlit-based dashboard provides a secure, interactive web platform. Users must log in with valid credentials to access the dashboard. Once

logged in, they can upload a single report at a time and view the predicted health condition, along with interactive visualizations such as probability charts, heatmaps, and feature importance plots. This setup ensures a seamless workflow from data collection and model training to single-report prediction and visualized output, making the system user-friendly and scalable for real-time health analysis.

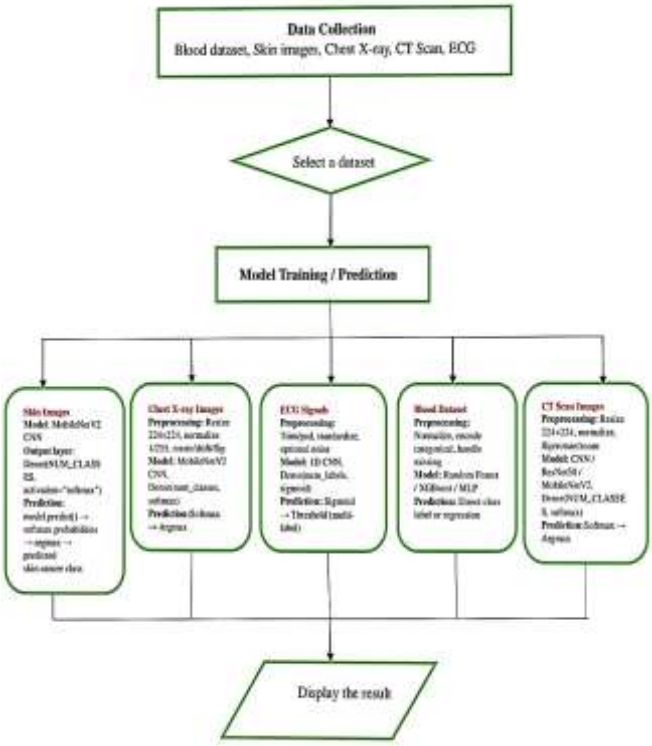

Figure 1: Workflow of AyuSense Prediction Model

The figure 1 illustrates the workflow of a complete machine-learning pipeline designed for analyzing different types of medical data, including blood datasets, skin images, chest X-ray images, CT scan images, and ECG signals. The process begins with data collection, where all these medical data sources are gathered. After collection, the user selects a specific dataset to work with. Once a dataset is chosen, the system proceeds to model training, where each type of data follows its own preprocessing steps and machine-learning model.

Skin images and chest X-ray images are processed using MobileNetV2 CNN models after resizing and normalizing the images, with predictions generated through softmax and converted to class labels using argmax. ECG signals are trimmed or padded, standardized, and optionally denoised before being fed into a 1D CNN, with predictions obtained through sigmoid activation followed by thresholding for multilabel classification.

Blood datasets undergo normalization, categorical encoding, and missing-value handling before being analyzed using models such as Random Forest, XGBoost, or MLP, producing either class labels or regression outputs. CT scan images are resized, normalized, and augmented through flipping, rotating, or zooming, and classified using models like CNN, ResNet50, or MobileNetV2, with final predictions again derived through softmax and argmax.

4.ALGORITHM

The proposed AyuSense Smart Health Analysis System employs a unified data-driven framework to analyze multimodal medical data including images, biosignals, and laboratory reports through

preprocessing, normalization, and feature extraction. Deep learning models such as MobileNetV2, ResNet50, and 1D-CNN, along with a Random Forest classifier, are utilized for disease prediction across different data types using transfer learning and optimized training. The trained models generate disease predictions with confidence scores, which are presented through an interactive dashboard to enable accurate, real-time, and automated health assessment.

General CNN Algorithm (Deep Learning)

Input

Labeled medical image dataset

$$\mathbf{D} = \{(\boldsymbol{X_i, y_i})\}_{i=1}^{N}$$

Output

Predicted disease class $\hat{\boldsymbol{y}}$

Algorithm Steps

1. Represent input image X as a tensor

$$\boldsymbol{X} \in \boldsymbol{R^{H\times W\times C}}$$

2. Apply convolution using learnable filters:

$$\boldsymbol{Z_{i,j,k}} = \sum \boldsymbol{X_{i+m,j+n,c}} \cdot \boldsymbol{K_{m,n,c,k}} + \boldsymbol{b_k}$$

3. Apply ReLU activation:

$$\boldsymbol{A} = \boldsymbol{max}(\mathbf{0}, \boldsymbol{Z})$$

4. Perform pooling to reduce spatial dimensions:

$$\boldsymbol{P} = \boldsymbol{max}(\boldsymbol{A})$$

5. Repeat convolution and pooling to extract hierarchical features.
6. Flatten feature maps into a vector $\boldsymbol{f}$.
7. Pass fff through fully connected layers:

$$\boldsymbol{h} = \boldsymbol{W}\,\boldsymbol{f} + \boldsymbol{b}$$

8. Apply Softmax to obtain class probabilities:

$$\hat{\boldsymbol{y}}_{\boldsymbol{i}} = \frac{\boldsymbol{e^{h_i}}}{\sum_j \boldsymbol{e^{h_j}}}$$

9. Train the network using backpropagation and gradient-based optimization.

3.6 ResNet-50 Algorithm (Deep Learning with Residual Learning)

Input

Preprocessed medical image X

Output

Disease class prediction Y

Algorithm Steps

1. Load ResNet-50 pre-trained on ImageNet.
2. Pass input through initial convolution and pooling layers.
3. For each residual block, compute:

$$\boldsymbol{Y} = \boldsymbol{F}(\boldsymbol{X}, \{\boldsymbol{W_i}\}) + \boldsymbol{X}$$

4. Use identity (skip) connections to preserve gradient flow.

5. Stack multiple residual blocks to learn deep representations.
6. Apply global average pooling.
7. Feed extracted features into a fully connected layer.
8. Apply Softmax activation for classification.
9. Fine-tune network parameters using medical dataset.

3.7 MobileNetV2 Algorithm (Lightweight Deep Learning Model)

Input

Medical image M

Output

Disease class label D

Algorithm Steps

1. Normalize input image M.
2. Apply depthwise convolution:

$$Z_d = X * K_d$$

3. Apply pointwise (1×1) convolution:

$$Z_p = Z_d * K_p$$

4. Use inverted residual blocks with linear bottleneck.
5. Extract efficient and compact feature representations.
6. Apply global average pooling.
7. Classify features using fully connected and Softmax layers.

3.8 One-Dimensional CNN Algorithm (ECG Signal Analysis)

Input

ECG signal

$$S = \{s_1, s_2, \ldots, s_n\}$$

Output

Cardiac condition label L

Algorithm Steps

1. Preprocess ECG signal by noise removal and normalization.
2. Apply 1D convolution:

$$Z_i = \sum s_{i+k} \cdot w_k + b$$

3. Apply ReLU activation.
4. Perform 1D max pooling.
5. Stack convolution layers to learn temporal patterns.
6. Flatten extracted features.
7. Apply fully connected layers.
8. Use Softmax or Sigmoid for classification.

Skin Cancer Classification Using Mobilenetv2

Input: Skin lesion image X ∈ ℝ^{224×224×3}

Output: Predicted class label ŷ ∈ {1,2,...,C} with confidence

Step 1: **Input Image Representation**

X ∈ ℝ^{224×224×3}

Step 2: **Dataset Mapping**

D = {(X_i, y_i)}_{i=1}^{N}, y_i ∈ {1,...,C}

Step 3: **Data Augmentation**

X'_i = T(X_i)

where T includes random flip, rotation, zoom

Step 4: **Image Preprocessing**

X_norm = X / 127.5 - 1

Step 5: **Feature Extraction (MobileNetV2)**

F = f_MobileNetV2(X_norm), F ∈ ℝ^{H×W×K}, H=W=7, K=1280

Step 6: **Global Average Pooling**

z_k = (1/(H·W)) Σ_{i=1}^{H} Σ_{j=1}^{W} F_{i,j,k}, z ∈ ℝ^K

Step 7: **Dropout Regularization**

z' = z ⊙ m, m ~ Bernoulli(p=0.6)

Step 8: **Fully Connected Layer**

o = W·z' + b, W ∈ ℝ^{C×K}, b ∈ ℝ^C

Step 9: **Softmax Classification**

P(y=c|X) = exp(o_c) / Σ_{k=1}^C exp(o_k)

ŷ = argmax_c P(y=c|X)

Step 10: **Loss Function**

L = -log(P(y_true|X))

Step 11: **Optimization (Adam)**

θ_{t+1} = θ_t - α · (m̂_t / (√v̂_t + ε))

Step 12: **Transfer Learning Phase**

∂L / ∂θ_base = 0 (base layers frozen)

Step 13: **Fine-Tuning Phase**

∂L / ∂θ_base ≠ 0 (selected base layers trainable)

Step 14: **Learning Rate Adjustment**

α_new = 0.5 × α_old

Step 15: **Model Selection & Early Stopping**

Model_best = argmax(Accuracy_val)

Step 16: **Prediction on New Image**

ŷ_test = argmax(f(X_test))

Confidence = max(P(y|X_test)) × 100%

Algorithm 1: Skin Cancer Classification using MobileNetV2

Step-Wise Working of Skin Cancer Classification using MobileNetV2 with Mathematical Formulation

---

Step 1: Input Image Representation

Each skin lesion image is represented as a 3-channel RGB image:

$$\boldsymbol{X} \in \boldsymbol{R^{224\times224\times3}}$$

Where:

- 224 × 224 → image resolution
- 3 → RGB color channels

Step 2: Dataset Mapping

The dataset is defined as:

$$\mathbf{D} = \{(\boldsymbol{X_i}, \boldsymbol{y_i})\}_{\boldsymbol{i=1}}^{\boldsymbol{N}}$$

Where:

- X_i → input image
- $y_i \in \{1,2,\ldots,C\}$ → class label
- C → number of skin cancer classes

Step 3: Data Augmentation

Augmented images are generated using random transformations:

$$\boldsymbol{X_i'} = \boldsymbol{T(X_i)}$$

Where transformation function T includes:

- Horizontal/vertical flipping
- Rotation
- Zooming

This increases data diversity and improves generalization.

Step 4: Image Preprocessing

MobileNetV2 preprocessing normalizes pixel values:

$$X_{norm} = \frac{X}{127.5} - 1$$

This scales pixel intensities to the range [−1,+1].

Step 5: Feature Extraction using MobileNetV2

The pretrained MobileNetV2 acts as a feature extractor:

$$F = f_{MobileNet\,V2}(X_{norm})$$

Where:

$$F \in R^{H \times W \times K}$$

Typically:

- H = 7, W = 7
- K=1280 feature maps

Step 6: Global Average Pooling

Each feature map is averaged spatially:

$$z_k = \frac{1}{H \times W} \sum_{i=1}^{H} \sum_{j=1}^{W} F_{i,j,k}$$

Resulting feature vector:

$$z \in R^K$$

This reduces dimensionality and prevents overfitting.

Step 7: Dropout Regularization

Dropout randomly deactivates neurons:

$$z' = z \odot m$$

Where:

- m∼Bernoulli(p)
- p=0.6 (keep probability)

This improves robustness.

Step 8: Fully Connected Layer

The dense layer computes class scores:

$$o = W z' + b$$

Where:

- $W \in R^{C \times K}$
- $b \in R^C$

Step 9: Softmax Classification

Softmax converts scores into probabilities:

$$P(y = c|X) = \frac{e^{o_c}}{\sum_{k=1}^{C} e^{o_k}}$$

Predicted class:

$$\hat{y} = \arg\max_{c} P(y = c|X)$$

Step 10: Loss Function

Sparse categorical cross-entropy loss is used:

$$L = -\log(P(y_{true}|X))$$

This penalizes incorrect class predictions.

Step 11: Optimization using Adam

Model parameters are updated using Adam optimizer:

$$\theta_{t+1} = \theta_t - \alpha \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon}$$

Where:

- α → learning rate
- $\hat{m}_t$, $\hat{v}_t$ → bias-corrected moment estimates

Step 12: Transfer Learning Phase

Initially, MobileNetV2 layers are frozen:

$$\frac{\partial L}{\partial \theta_{base}} = 0$$

Only the classification layers are trained.

Step 13: Fine-Tuning Phase

Later, selected base layers are unfrozen:

$$\frac{\partial L}{\partial \theta_{base}} \neq 0$$

This allows the network to adapt features for skin cancer detection.

Step 14: Learning Rate Adjustment

Learning rate reduction is applied as:

$$\alpha_{new} = 0.5 \times \alpha_{old}$$

This stabilizes training.

Step 15: Model Selection and Early Stopping

Best model is selected based on validation accuracy:

$$Model_{best} = \arg\max(Accuracy_{val})$$

Training stops when validation loss no longer improves.

Step 16: Prediction on Test Image

For a new image X_{test} :

$$\hat{y}_{test} = \arg\max(f(X_{test}))$$

Confidence score:

$$Confidence = max(P(y|X_{test})) \times 100\%$$

The proposed system formulates skin cancer classification as a multi-class optimization problem using transfer learning and fine-tuning. By integrating MobileNetV2 feature extraction, global average pooling, dropout regularization, and softmax classification, the model achieves efficient and accurate skin cancer diagnosis.

Chest X-ray Disease Classification using MobileNetV2

Input: Chest X-ray image $X \in \mathbb{R}^{224\times224\times3}$

Output: Predicted class label $\hat{y} \in$ {COVID-19, NORMAL, PNEUMONIA} with confidence

Step 1: **Input Image Representation**

$X \in \mathbb{R}^{224\times224\times3}$

Step 2: **Dataset Formation**

$D = \{(X_i, y_i)\}_{i=1}^{N},\ y_i \in \{1,2,3\}$

Step 3: **Data Augmentation**

$X'_i = T(X_i)$

where T includes rotation, width/height shift, shear, zoom, horizontal flip

Step 4: **Image Normalization**

$X_{norm} = X / 255$

Step 5: **Feature Extraction (MobileNetV2)**

$F = f_{MobileNetV2}(X_{norm}),\ F \in \mathbb{R}^{H\times W\times K},\ H=W=7,\ K=1280$

$\partial L / \partial \theta_{base} = 0$ (base layers frozen initially)

Step 6: **Global Average Pooling**

$z_k = (1/(H\cdot W)) \Sigma_{i=1}^{H} \Sigma_{j=1}^{W} F_{i,j,k},\ z \in \mathbb{R}^K$

Step 7: **Dropout Regularization**

$z' = z \odot m,\ m \sim Bernoulli(p=0.7)$

Step 8: **Fully Connected Layer**

$h = ReLU(W_1 \cdot z' + b_1),\ W_1 \in \mathbb{R}^{128\times K},\ b_1 \in \mathbb{R}^{128}$

Step 9: **Output Classification Layer**

$o = W_2 \cdot h + b_2,\ W_2 \in \mathbb{R}^{C\times128},\ C=3$

Step 10: **Softmax Probability Estimation**

$P(y=c|X) = \exp(o_c) / \Sigma_{k=1}^{C} \exp(o_k)$

$\hat{y} = \text{argmax}_c\ P(y=c|X)$

Step 11: **Loss Function**

$L = - \Sigma_{c=1}^{C} y_c \log(P(y=c|X))$

Step 12: **Optimization (Adam)**

$\theta_{t+1} = \theta_t - \alpha \cdot (\hat{m}_t / (\sqrt{\hat{v}_t} + \varepsilon))$

Step 13: **Fine-Tuning Phase**

$\partial L / \partial \theta_{base} \neq 0$ (selected deeper base layers trainable)

Step 14: **Learning Rate Adjustment**

$\alpha_{new} = 0.5 \times \alpha_{old}$

Step 15: **Model Selection & Early Stopping**

$Model_{best} = \text{argmax}(Accuracy_{val})$

Step 16: **Prediction on New X-ray Image**

$\hat{y}_{test} = \text{argmax}(f(X_{test}))$

$Confidence = \max(P(y|X_{test})) \times 100\%$

Algorithm 2: Chest X-ray Disease Classification using MobileNetV2

Step-Wise Working of Chest X-ray Disease Classification System Using MobileNetV2 and Transfer Learning

---

Step 1: Input Image Representation

Each chest X-ray image is represented as a 3-channel RGB image after resizing:

$$X \in R^{224\times224\times3}$$

Where:

- 224×224 → spatial resolution

- 3→ RGB channels

Step 2: Dataset Formation

The dataset is organized into training, validation, and testing sets:

$$\mathbf{D} = \{(\boldsymbol{X_i}, \boldsymbol{y_i})\}_{i=1}^{N}$$

Where:

- Xi → chest X-ray image
- $y_i \in \{1,2,3\}$ → class label
  - COVID-19
  - NORMAL
  - PNEUMONIA

Step 3: Data Augmentation

To improve model generalization, real-time data augmentation is applied:

$$\boldsymbol{X_i'} = \boldsymbol{T}(\boldsymbol{X_i})$$

Where transformation function T includes:

- Rotation
- Width and height shifting
- Shearing
- Zooming
- Horizontal flipping

This simulates real-world variations in X-ray acquisition.

Step 4: Image Normalization

All pixel values are normalized using min-max scaling:

$$\boldsymbol{X_{norm}} = \frac{\boldsymbol{X}}{\mathbf{255}}$$

This scales pixel intensities to the range [0,1], ensuring numerical stability during training.

Step 5: Feature Extraction using MobileNetV2

A pretrained MobileNetV2 network is used as a feature extractor:

$$\boldsymbol{F} = \boldsymbol{f_{MobileNet\,V2}}(\boldsymbol{X_{norm}})$$

Where:

$$\boldsymbol{F} \in \boldsymbol{R}^{H \times W \times K}$$

Typical dimensions:

- H=7, W=7
- K=1280 feature channels

Initially, all MobileNetV2 layers are frozen:

$$\frac{\boldsymbol{\partial L}}{\boldsymbol{\partial \theta_{base}}} = \mathbf{0}$$

Step 6: Global Average Pooling

Spatial feature maps are reduced using Global Average Pooling:

$$\boldsymbol{z_k} = \frac{\mathbf{1}}{\boldsymbol{H \times W}} \sum_{\boldsymbol{i=1}}^{\boldsymbol{H}} \sum_{\boldsymbol{j=1}}^{\boldsymbol{W}} \boldsymbol{F_{i,j,k}}$$

Resulting feature vector:

$$z \in R^K$$

This reduces model complexity and prevents overfitting.

Step 7: Dropout Regularization

Dropout is applied to reduce co-adaptation of neurons:

$$z' = z \odot m$$

Where:

- m ~ Bernoulli(p)
- p=0.7 (keep probability)

Step 8: Fully Connected Layer

A dense layer with ReLU activation is applied:

$$h = ReLU(W_1 z' + b_1)$$

Where:

- $W_1 \in R^{128 \times K}$
- $b_1 \in R^{128}$

Step 9: Output Classification Layer

The final dense layer computes class scores:

$$o = W_2 h + b_2$$

Where:

- $W_2 \in R^{C \times 128}$
- C=3 classes

Step 10: Softmax Probability Estimation

Softmax converts scores into probabilities:

$$P(y = c|X) = \frac{e^{o_c}}{\sum_{k=1}^{C} e^{o_k}}$$

Predicted class:

$$\hat{y} = \arg\max_c P(y = c|X)$$

Step 11: Loss Function

Categorical cross-entropy loss is used:

$$L = -\sum_{c=1}^{C} y_c \log(P(y = c|X))$$

This penalizes incorrect predictions across all classes.

Step 12: Optimization using Adam Optimizer

Model parameters are updated as:

$$\theta_{t+1} = \theta_t - \alpha \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon}$$

Where:

- α → learning rate
- $\hat{m}_t, \hat{v}_t$ → bias-corrected moment estimates

Step 13: Fine-Tuning Phase

In fine-tuning, selected deeper layers of MobileNetV2 are unfrozen:

$$\frac{\partial L}{\partial \theta_{base}} \neq 0$$

Training continues with a smaller learning rate to refine domain-specific features.

Step 14: Learning Rate Adjustment

Learning rate reduction is applied as:

$$\alpha_{new} = 0.5 \times \alpha_{old}$$

This improves convergence when validation loss plateaus.

Step 15: Model Selection and Early Stopping

Best model is selected based on validation accuracy:

$$Model_{best} = \arg\max(Accuracy_{val})$$

Training is stopped early when validation loss no longer improves.

Step 16: Prediction on New Chest X-ray Image

For a new input image X_{test}:

$$\hat{y}_{test} = \arg\max(f(X_{test}))$$

Confidence score:

$$Confidence = max\left(P(y|X_{test})\right) \times 100\%$$

The proposed chest X-ray classification system formulates disease detection as a multi-class optimization problem using transfer learning and fine-tuning. By integrating MobileNetV2 feature extraction, global average pooling, dropout regularization, and softmax classification, the system achieves accurate and computationally efficient diagnosis of COVID-19, pneumonia, and normal cases.

CT Scan Disease Classification System using ResNet50

Input: CT scan image X ∈ ℝ^{224×224×3}

Output: Predicted disease class ŷ ∈ {1,2,…,C}

Step 1: **Input Image Representation**

X ∈ ℝ^{224×224×3}

Step 2: **Dataset Formation**

D = {(X_i, y_i)}_{i=1}^N, y_i ∈ {1,…,C}

Step 3: **Data Augmentation**

X'_i = T(X_i), T ∈ {flip, rotate, color jitter}

Step 4: **Image Preprocessing**

X_norm = (X - μ)/σ, μ=[0.485,0.456,0.406], σ=[0.229,0.224,0.225]

Step 5: **Feature Extraction (ResNet50)**

F = f_ResNet50(X_norm), F ∈ ℝ^{7×7×2048}

Step 6: **Residual Blocks**

Y = ReLU(X + f(X,W))

Step 7: **Global Average Pooling**

z_k = (1/(H*W)) ∑_{i=1}^{H} ∑_{j=1}^{W} F_{i,j,k}

z ∈ ℝ^{2048}

Step 8: **Dropout Regularization**

z' = Dropout(z), rate=0.4

Step 9: **Fully Connected Layer**

o = W z' + b, W ∈ ℝ^{C×2048}, b ∈ ℝ^{C}

Step 10: **Softmax Classification**

P(y=c|X) = e^{o_c} / ∑_{k=1}^{C} e^{o_k}

ŷ = argmax_c P(y=c|X)

Step 11: **Loss Function**

L_CE = - ∑_{i=1}^{C} y_i log(P(y=i|X))

Step 12: **Optimization (Adam)**

θ_{t+1} = θ_t - α (m̂_t / (√v̂_t + ε)), α=1e-4

Step 13: **Transfer Learning Phase**

Freeze all ResNet50 layers except classifier:

∂L/∂θ_base = 0

Step 14: **Fine-Tuning Phase**

Unfreeze last residual blocks:

∂L/∂θ_base ≠ 0

Step 15: **Training Control**

Model Checkpoint → save best validation accuracy

Learning Rate Scheduler: LR_new = 0.1 * LR_old

Early Stopping if validation loss stagnates

Step 16: **Prediction on New CT Scan Image**

ŷ_test = argmax(Softmax(f_ResNet50(X_new)))

Confidence = max(P(y|X_new)) × 100%

Algorithm 3: CT Scan Disease Classification System using ResNet50

Mathematical Model and Working of CT Scan Disease Classification System (ResNet50)

Step 1: Problem Definition

Given a **CT scan image**, classify it into one of **C disease classes**:

$$\boldsymbol{X} \in \boldsymbol{R^{224\times224\times3}}, \hat{\boldsymbol{y}} \in \{\boldsymbol{1, 2, \dots, C}\}$$

Where:

C=number of disease categories (Bone Break, Bra in Tumor, Lung Cancer, Renal Malignancy, Skin Lesions)

Step 2: Dataset Representation

Training dataset:

$$\mathbf{D} = \{(\boldsymbol{X_i, y_i})\}_{\boldsymbol{i=1}}^{\boldsymbol{N}}, \hat{\boldsymbol{y}} \in \{\boldsymbol{1, 2, \dots, C}\}$$

- X_i → input CT scan image
- y_i → class label (integer encoded)

Step 3: Data Augmentation

Random transformations to improve generalization:

$$\boldsymbol{X_i'} = \boldsymbol{T(X_i)}$$

Where T includes:

- Random horizontal flip
- Random rotation
- Color jitter (brightness, contrast, saturation)

Step 4: Image Preprocessing

Normalize pixel values using ImageNet statistics:

$$\boldsymbol{X_{norm}} = \frac{\boldsymbol{X - \mu}}{\boldsymbol{\sigma}}$$

μ=[0.485,0.456,0.406], σ=[0.229,0.224,0.225]

Step 5: Feature Extraction (ResNet50 Base)

Pretrained ResNet50 extracts deep features:

$$F = f_{ResNet50}(X_{norm}), \quad F \in R^{7\times7\times2048}$$

Step 6: Residual Blocks (Convolution + Shortcut)

Each block:

$$Y = ReLU\left(X + f(X, W)\right)$$

Where:

- $f(X, W)$ → convolution + batch norm + activation
- Shortcut connection ensures gradient flow

Step 7: Global Average Pooling

Reduces spatial dimensions:

$$z_k = \frac{1}{H \times W} \sum_{i=1}^{H} \sum_{j=1}^{W} F_{i,j,k}, \quad z \in R^{2048}$$

Step 8: Dropout Regularization

Randomly disables neurons to prevent overfitting:

$$z' = Dropout(z), \qquad dropout\ rate = 0.4$$

Step 9: Fully Connected Layer

Compute logits for each class:

$$o = Wz' + b, \quad W \in R^{C\times2048}, b \in R^{C}$$

Step 10: Softmax Classification

Convert logits into probabilities:

$$P(y = c|X) = \frac{e^{o_c}}{\sum_{k=1}^{C} e^{o_k}},$$

$$\hat{y} = \arg\max_{c} P(y = c|X)$$

Step 11: Loss Function (Cross-Entropy)

$$L_{CE} = -\sum_{c=1}^{C} y_i \log(P(y = i|X))$$

Where y_i =1 for true class, 0 otherwise.

Step 12: Optimization (Adam)

Model parameters updated using Adam:

$$\theta_{t+1} = \theta_t - \alpha \frac{\widehat{m_t}}{\sqrt{\widehat{v_t}} + \epsilon}, \quad \alpha = 1 \times 10^{-4}$$

Step 13: Transfer Learning Phase

Initially, **all ResNet50 layers frozen except classifier**:

$$\frac{\partial L}{\partial \theta_{base}} = 0$$

- Only final fully connected layer is trained.

Step 14: Fine-Tuning Phase

Later, **last residual blocks unfrozen**:

$$\frac{\partial L}{\partial \theta_{base}} \neq 0$$

- Adapts ImageNet features to CT scan images.

Step 15: Training Control Mechanisms

- **Model Checkpoint:** Save model when validation accuracy ↑
- **Learning Rate Scheduler:** LRnew =0.1×LR_{old}
- **Early Stopping:** Stop if validation loss stops improving

---

Step 16: Prediction on New CT Scan Image

For new image X_{new}:

$$\hat{y} = arg\ max(Softmax(f_{ResNet50}(X_{new})))$$

Confidence:

$$Confidence = max(P(y|X_{test})) \times 100\%$$

The proposed CT scan disease classification system leverages ResNet50-based transfer learning to effectively identify multiple medical conditions such as Bone Break, Brain Tumor, Lung Cancer, Renal Malignancy, and Skin Lesions. By integrating deep residual feature extraction, global average pooling, dropout regularization, and softmax classification, the system is able to capture complex spatial and textural patterns in CT images while reducing overfitting and computational complexity. The two-phase training strategy, consisting of transfer learning and fine-tuning, ensures that pretrained ImageNet features are effectively adapted to domain-specific CT scan data. Coupled with data augmentation, learning rate scheduling, model checkpointing, and early stopping, the model achieves high accuracy, robustness, and generalization on unseen images. This framework provides a practical and reliable solution for real-world diagnostic support, enabling healthcare professionals to make informed decisions based on automated CT scan analysis.

ECG Disease Classification using 1D-CNN and Multi-Label Learning

Input: 12-lead ECG signal X ∈ ℝ^(12×T) (T = 1000 samples)

Output: Predicted cardiac condition probabilities Ŷ ∈ {0,1}^71

Step 1: **Signal Preprocessing**

1.1 Standardize signal length: pad or truncate X to T samples

1.2 Normalize each lead: X_norm = (X - μ) / (σ + ε)

Step 2: **Data Augmentation (Training Phase)**

2.1 Add Gaussian noise: X_aug = X_norm + λ * N(0,1)

Step 3: **1D Convolutional Feature Extraction**

3.1 Conv1D Layer 1: F_1 = ReLU(Conv1D(X_aug, W_1) + b_1)

3.2 Batch Normalization: ˆF_1 = (F_1 - μ_F) / √(σ_F^2 + ε)

3.3 Max Pooling: P_1 = max(F_1[i:i+k])

3.4 Conv1D Layer 2: F_2 = ReLU(Conv1D(P_1, W_2) + b_2)

3.5 Conv1D Layer 3: F_3 = ReLU(Conv1D(F_2, W_3) + b_3)

Step 4: **Global Average Pooling**

4.1 z_k = (1/T') ∑_{t=1}^{T'} F_{3,k}(t)

4.2 Resulting feature vector z ∈ ℝ^256

Step 5: **Fully Connected Layer**

5.1 h = ReLU(W * z + b)

5.2 Apply Dropout: h' = h ⊙ m, m ~ Bernoulli(0.6)

Step 6: **Multi-Label Output Layer**

6.1 o = W_o * h' + b_o

6.2 Apply Sigmoid: P(y_j=1|X) = 1 / (1 + exp(-o_j)), j=1…71

Step 7: **Loss and Optimization**

7.1 Compute Binary Cross-Entropy Loss:

L = -(1/N) ∑_{i=1}^N ∑_{j=1}^{71} [y_{ij} log(p_{ij}) + (1-y_{ij}) log(1-p_{ij})]

7.2 Update weights using Adam optimizer:

θ_{t+1} = θ_t - α * (m̂_t / (√(v̂_t) + ε)), α=10^-4

Step 8: **Prediction**

8.1 For test signal X_test, compute probabilities Ŷ = P(y|X_test)

8.2 Rank top-k conditions with highest probabilities

Algorithm 4: ECG Disease Classification using 1D-CNN and Multi-Label Learning

Step-Wise Working of ECG Disease Classification System PTB-XL Dataset using 1D CNN and Multi-Label Learning

1. Problem Definition

Given a **12-lead ECG signal**, the objective is to predict **multiple cardiac conditions simultaneously** from a set of **71 SCP diagnostic labels**.

Input

$$\boldsymbol{X} \in \boldsymbol{R^{12 \times T}}$$

where

- 12 → ECG leads
- T=1000 samples (10 seconds at 100 Hz)

Output

$$\boldsymbol{\hat{Y} = \{0,1\}^{71}}$$

where each output represents the presence probability of a cardiac condition.

2. Dataset Representation (PTB-XL)

The dataset is represented as:

$$\mathbf{D} = \{(\boldsymbol{X_i, y_i})\}_{i=1}^{N}$$

Where:

- X_i → ECG signal
- Y_i → multi-label target vector
- Labels are encoded using **Multi-Label Binarization**:

$$\boldsymbol{Y_{bin} = MLB(Y)}$$

3. Signal Length Standardization

Each ECG signal is padded or truncated to fixed length:

$$X_i = \begin{cases} X_i[0:T] & if\ T_i > T \\ pad\ (X_i)\ if\ T_i < T \end{cases}$$

This ensures uniform input dimensions for CNN processing.

---

4. Signal Normalization

Each ECG lead is normalized:

$$X_{norm} = \frac{X - \mu}{\sigma}$$

Where:

- μ → mean of signal
- σ → standard deviation
- $\epsilon = 10^{-6}$ avoids division by zero

---

5. Data Augmentation (Noise Injection)

Gaussian noise is added during training:

$$X' = X_{norm} + \lambda . N(0,1)$$

Where:

- λ=0.01 controls noise strength

This improves robustness to real-world ECG noise.

---

6. Convolutional Feature Extraction (1D CNN)

6.1 First Convolution Layer

$$F_1 = ReLU(Conv1D(X^1, W_1) + b_1)$$

Kernel size = 7

Filters = 64

6.2 Batch Normalization

$$\hat{F} = \frac{F - \mu_F}{\sqrt{\sigma_F^2 + \epsilon}}$$

Stabilizes training and accelerates convergence.

6.3 Max Pooling

$$P = max(F_{i:i+k})$$

Downsamples temporal resolution by factor of 2.

6.4 Deeper Convolution Layers

Second convolution:

$$F_2 = ReLU(Conv1D(P, W_2) + b_2)$$

- Filters = 128
- Kernel size = 5

Third convolution:

$$F_3 = ReLU(Conv1D(P, W_3) + b_3)$$

- Filters = 256
- Kernel size = 3

---

7. Global Average Pooling (Temporal Aggregation)

$$z_k = \frac{1}{T'} \sum_{t=1}^{T'} F_{3,k}(t)$$

Result:

$$z \in R^{256}$$

This removes temporal dependency and reduces parameters.

---

8. Fully Connected Layer

$$h = ReLU(W_z + b)$$

Where:

- $W \in R^{256\times256}$

---

9. Dropout Regularization

$$h' = h \odot m$$

$$where\ m \sim \text{Bernoulli}(0.6)$$

Prevents overfitting.

---

10. Multi-Label Output Layer

$$o = W_o h^1 + b_o$$

Final activation using **Sigmoid**:

for j=1…71

---

11. Loss Function (Binary Cross-Entropy)

$$L = -\frac{1}{N} \sum_{i=1}^{N} \sum_{j=1}^{71} [y_{ij} log(p_{ij}) + (1 - y_{ij}) log(1 - p_{ij})]$$

Suitable for **multi-label classification**.

---

12. Optimization using Adam

$$\theta_{t+1} = \theta_t - \alpha \frac{\widehat{m_t}}{\sqrt{\widehat{v_t}} + \epsilon}$$

Learning rate:

$$\alpha = 1 \times 10^{-4}$$

---

13. Model Evaluation Metrics

Accuracy

$$Accurracy = \frac{TP + TN}{TP + TN + FP + FN}$$

AUC

$$AUC = \int_0^1 TPR(FPR) d(FPR)$$

F1 Score

$$F1 = \frac{2TP}{2TP + FP + FN}$$

---

14. ECG Signal Analysis (Rule-Based)

R-Peak Detection

$$R - peaks = find\_peaks(X_{bandpass})$$

RR Interval

$$RR_i = \frac{R_{i+1} - R_i}{f_s}$$

Heart Rate

$$HR = \frac{60}{RR}$$

---

15. QRS Duration Estimation

$$QRS_{ms} = \frac{width\ samples}{f_s} \times 1000$$

---

16. ST-Segment Deviation

$$\Delta ST = ST_{value} - Baseline$$

Threshold:

$$|\Delta ST| > 0.12 \times \sigma_{signal}$$

---

17. T-Wave Inversion Detection

T < 0 ⇒ Inversion

---

18. PVC Detection

PVC is detected if:

$$RR_i < 0.8 \times RR_{i-1}$$

and waveform correlation:

$$corr < 0.85$$

---

19. Severity and Confidence Estimation

$$Severity = \begin{cases} \text{Severe} & p > 0.85 \\ \text{Moderate} & p > 0.65 \\ \text{Mild} & p > 0.45 \\ \text{Low} & \text{otherwise} \end{cases}$$

---

20. Final Prediction

$$\hat{y} = \arg\max_j P(y_j | X)$$

Top-k conditions are reported with probabilities

The proposed ECG diagnostic system formulates cardiac condition detection as a multi-label classification problem using deep 1D convolutional neural networks. By combining signal preprocessing, temporal feature extraction, global pooling, and sigmoid-based probability estimation, the system accurately predicts 71 ECG abnormalities. The integration of rule-based ECG analysis further enhances interpretability and clinical relevance.

Smart Health Lab Report Disease Prediction using Random Forest

Input: Patient lab report R (PDF/Image) containing blood test parameters

Output: Predicted disease class ŷ_new with confidence

Step 1: **Automatic Text Extraction**

1.1 If PDF: T = Σ_{p=1}^{P} extract_text(p)

1.2 If Image: T = Tesseract(I)

Step 2: **Feature Extraction**

2.1 For each parameter x_i (i = 1…21):

x_i = float(regex(T)) if found, else NaN

2.2 Form feature vector: X_new = [x_1, x_2, …, x_21]

Step 3: **Feature Preprocessing**

3.1 Standardization: X_scaled = (X_new - μ) / σ

Step 4: **Random Forest Prediction**

4.1 Each decision tree T_j predicts: ŷ_j = T_j(X_scaled)

4.2 Ensemble prediction via majority voting:

ŷ_new = mode(ŷ_1, ŷ_2, …, ŷ_M)

4.3 Class probabilities:

P(y=c | X_new) = (1/M) Σ_{j=1}^{M} I(ŷ_j = c)

Step 5: **Confidence Estimation**

5.1 Confidence = max_c P(y=c | X_new) × 100

Step 6: **Interpretation**

6.1 Provide clinical reasoning based on parameter patterns:

- High WBC → Infection
- Low HGB / RBC → Anemia
- Abnormal PLT → Clotting disorder

Algorithm 5: Smart Health Lab Report Disease Prediction using Random Forest

Mathematical Model and Working of Smart Health Lab Report Disease Prediction System

---

1. Problem Definition

The objective of the system is to predict a disease class based on blood laboratory parameters automatically extracted from medical reports (PDF/Image).

Input

$$X = [x_1, x_2, \ldots, x_n] \in R^n$$

Where:

- n=21 laboratory features (WBC, RBC, HGB, HCT, MCV, MCH, PLT, etc.)

Output

$$\hat{y} \in \{D_1, D_2, \ldots, D_k\}$$

Where:

- k = number of disease classes
- $\hat{y}$ = predicted disease

---

2. Dataset Representation

The dataset is represented as:

$$D = \{(X_i, y_i)\}_{i=1}^{N}$$

Where:

- $X_i \in R^{21}$ → blood parameters
- y_i → disease label

3. Label Encoding

Since disease labels are categorical, they are converted into numeric form using Label Encoding:

$$y_{enc} = f(y),$$

$$f : y \rightarrow \{0, 1, 2, \dots, k-1\}$$

This enables machine learning algorithms to process the target variable.

4. Train–Test Split

The dataset is divided as:

$$D_{train} = 80\%, \quad D_{test} = 20\%,$$

$$(X_{train}, X_{test}, Y_{train}, y_{test})$$

This ensures unbiased model evaluation.

5. Feature Standardization

Numerical features are normalized using StandardScaler:

$$X_{scaled} = \frac{X - \mu}{\sigma}$$

Where:

- μ → mean of feature
- σ → standard deviation

Standardization ensures that features contribute equally to the learning process.

6. Random Forest Classifier

A Random Forest is an ensemble of decision trees.

6.1 Decision Tree Learning

Each decision tree T_j is trained on a bootstrap sample:

$$D_j \sim Bootstrap(D_{train})$$

6.2 Feature Randomness

At each split, a random subset of features is selected:

$$F_j \subseteq F, \quad |F_j| = \sqrt{n}$$

This reduces correlation between trees.

6.3 Gini Impurity (Split Criterion)

The quality of splits is measured using Gini Impurity:

$$G = 1 - \sum_{c=1}^{k} p_c^2$$

Where:

- p_c = probability of class c at the node

The split minimizing G is selected.

7. Ensemble Prediction

Each tree produces a prediction:

$$\hat{y}_J = T_J(X)$$

The final prediction is obtained by majority voting:

$$\hat{y} = mode(\hat{y_1}, \hat{y_2}, \dots, \hat{y_{200}})$$

---

8. Probability Estimation

Class probabilities are calculated as:

$$P(y = c \mid X) = \frac{1}{M} \sum_{j=1}^{M} I(\hat{y}_J = c)$$

Where:

- M=200 trees
- I is indicator function

---

9. Model Pipeline

The complete pipeline is mathematically represented as:

$$\hat{y} = f_{RF}\left(\frac{X - \mu}{\sigma}\right)$$

Where:

- f_{RF} → trained Random Forest classifier

---

10. Model Evaluation Metrics

Accuracy

$$Accurracy = \frac{TP + TN}{TP + TN + FP + FN}$$

Precision

$$Precision = \frac{TP}{TP+FP}$$

Recall

$$Recall = \frac{TP}{TP+FN}$$

F1-Score

$$F1 = \frac{2.Precision.Recall}{Precision + Recall}$$

---

11. Model Persistence

The trained model and label encoder are saved as:

$$Model \rightarrow disease\_model.pkl$$

Ensuring reproducibility and deployment readiness.

---

12. Automatic Text Extraction (OCR / PDF)

PDF Extraction

$$T = \sum_{p=1}^{P} extract\_text(p)$$

Image OCR

$$T = Tesseract(I)$$

Where:

- T → extracted text

---

13. Feature Extraction Using Regular Expressions

Each lab parameter is extracted as:

$$x_i = \begin{cases} float(regex(T)), & if\ found \\ NaN, & otherwise \end{cases}$$

---

14. Prediction on New Patient Data

The extracted feature vector is:

$$X_{new} = [x_1, x_2, \ldots, x_{21}]$$

Prediction:

$$\widehat{y_{new}} = f_{RF}(X_{new})$$

---

15. Confidence Estimation

$$Confidence = \max_c P(y = c \mid X_{new}) \times 100$$

---

16. Explanation of Prediction (Clinical Basis)

The prediction is interpreted based on medical correlations:

- High WBC → Infection or inflammation
- Low HGB / RBC → Anemia
- Abnormal PLT indices → Clotting disorders

The **combined pattern** of all parameters determines the final disease classification.

The proposed Smart Health Lab Report Prediction System integrates automatic report parsing, feature normalization, and a Random Forest ensemble classifier to accurately identify diseases from blood test parameters. The system eliminates manual data entry, improves diagnostic efficiency, and provides interpretable predictions with confidence scores, making it suitable for real-time clinical decision support.

CNN pipeline for Skin Disease Classification

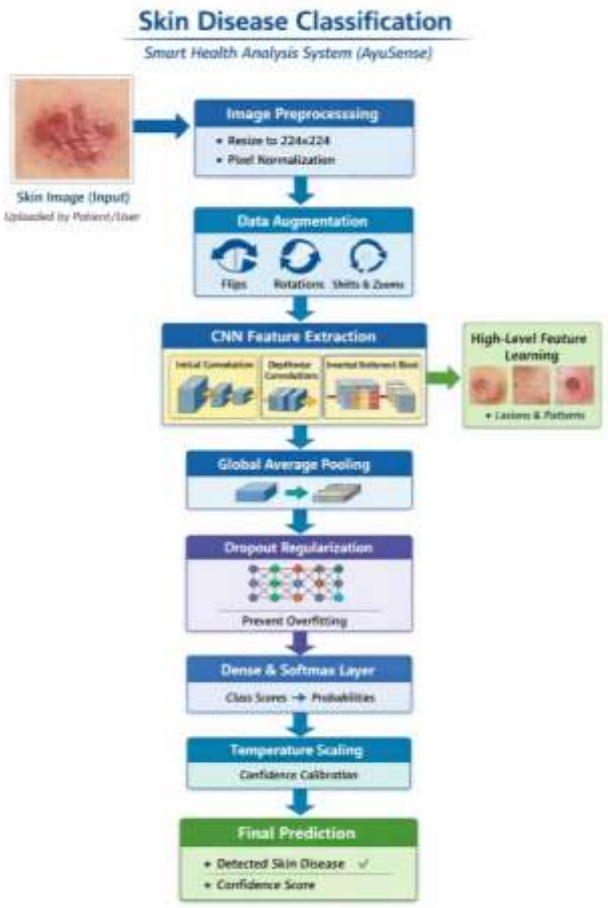

Figure 2: CNN working model for Skin Disease Classification

The Skin Disease Classification module in the AyuSense Smart Health Analysis System begins

with the user uploading a skin image through the application. The image is first preprocessed by resizing it to a fixed dimension of 224 × 224 pixels and normalizing pixel values to ensure consistency and reduce noise. To improve model robustness and avoid overfitting, data augmentation techniques such as image flipping, rotation, shifting, and zooming are applied. The enhanced images are then fed into a Convolutional Neural Network (CNN), where multiple layers including initial convolutions, depthwise convolutions, and inverted bottleneck blocks extract meaningful features such as lesions, textures, and skin patterns. High-level features learned by the CNN are condensed using global average pooling, followed by dropout regularization to further prevent overfitting. The processed features are passed through dense layers and a Softmax classifier to generate class probabilities. Temperature scaling is finally applied to calibrate the confidence scores, and the system outputs the detected skin disease along with a reliable confidence score.

1D CNN pipeline for ECG Signal Analysis

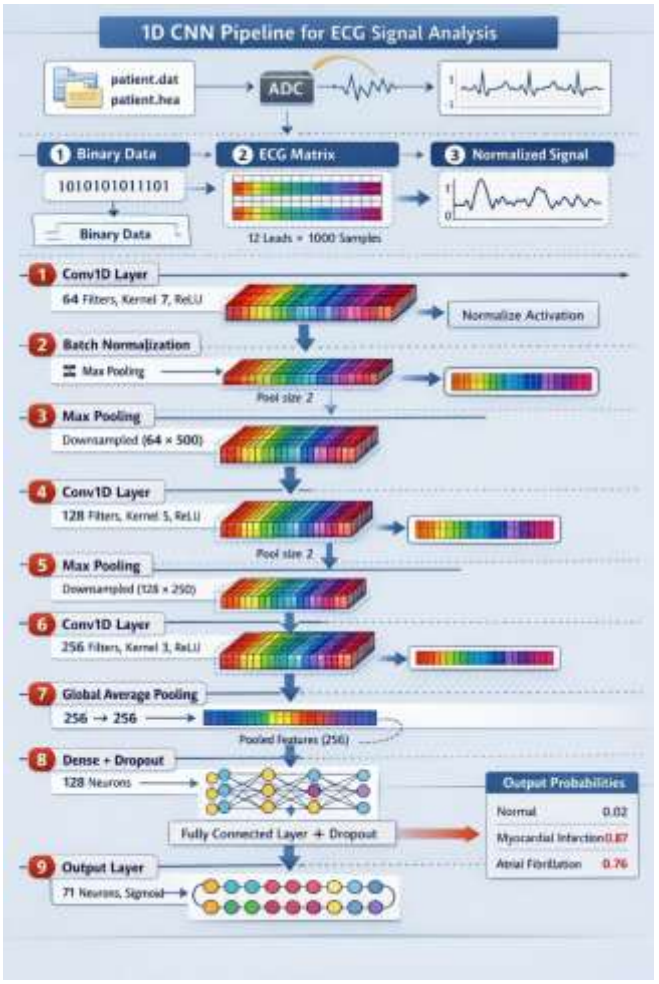

Figure 2: 1D CNN pipeline for ECG Signal Analysis

The 1D CNN pipeline for ECG signal analysis in the AyuSense Smart Health Analysis System begins with the acquisition of raw ECG data files, which are converted from binary format into digital signals using an analog-to-digital converter (ADC). The ECG signals are organized into a matrix representing multiple leads and samples, and the signals are then normalized to ensure consistent amplitude ranges. The normalized ECG data is passed through a sequence of one-dimensional convolutional layers that use increasing numbers of filters and decreasing kernel sizes to capture both low-level and high-level temporal features of the heart signal. Batch normalization and max pooling layers are applied after convolution to stabilize

learning, reduce noise, and downsample the signal while preserving important waveform characteristics. As the network deepens, it learns discriminative patterns related to different cardiac conditions. Global average pooling is used to condense the extracted feature maps into a fixed-length feature vector, which is then processed by a fully connected dense layer with dropout to prevent overfitting. Finally, the output layer with sigmoid activation produces probability scores for multiple heart conditions, such as normal rhythm, myocardial infarction, and atrial fibrillation, providing reliable diagnostic predictions.

5. RESULTS AND SIMULATION

The **AyuSense** Smart Health Analysis System successfully demonstrated accurate performance across all its diagnostic modules. The Blood Report Analysis module correctly extracted CBC values from uploaded reports and predicted the user's health status with clear probability scores, identifying the sample case as Healthy with the highest confidence. The CT Scan and X-ray modules showed excellent accuracy by detecting a bone fracture with 99.95% confidence, proving the strength of the deep-learning models in identifying patterns, textures, and abnormalities in medical images. The system's dashboard worked smoothly, allowing easy navigation between Blood Test, ECG, Skin Disease, CT, and X-ray modules. Overall, the results confirm that AyuSense provides reliable, fast, and accurate multi-modal health predictions through an integrated AI-based platform, effectively supporting early diagnosis and improving healthcare accessibility.

Figure 4: Login Page

The **Figure 4** depicts a login interface for AyuSense, a Smart Health Analysis System. It features a dark-themed design with the system name and description centered at the top. Below that, there are options to select an action between "Login" and "Sign Up," with "Login" currently selected.

The login form includes fields for entering a username and password, with the username field pre-filled with "Admin." The password field obscures the input for security, and there is an eye icon to toggle password visibility. At the bottom of the form, there is a login button featuring a lock icon and the text "Login." The overall interface is clean and user-friendly, designed to facilitate secure access to the system.

Figure 5: Dashboard

The **Figure 5** shows the dashboard of AyuSense, a Smart Health Analysis System, after the admin has logged in. The interface welcomes the admin and offers a logout option. It presents five different health analysis modules for selection: Blood Test, ECG (Electrocardiogram), Skin condition detection, CT Scan image analysis, and Chest X-ray examination. Each module has a brief description and an associated button to start the respective analysis, allowing users to easily navigate and access various diagnostic tools within the system. The overall design is clean and organized with distinct color-coded tiles for each module.

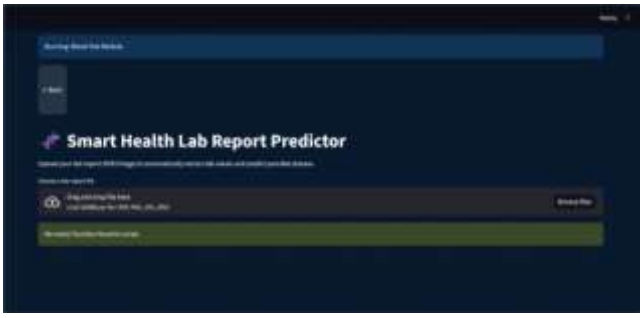

Figure 6: Blood Report Prediction

The **Figure 6** shows the Blood Test Module of the Smart Health Lab Report Predictor within the AyuSense system. This module allows users to upload lab reports in PDF or image formats (PDF, PNG, JPG, JPEG) with a file size limit of 200MB to automatically extract lab values and predict possible diseases. The interface includes a drag-and-drop area for file uploading, as well as a button to browse files manually. A "Back" button is available for navigation, and there is a message displayed at the bottom stating "No main() function found in script." The overall design follows a dark theme consistent with the rest of the system.

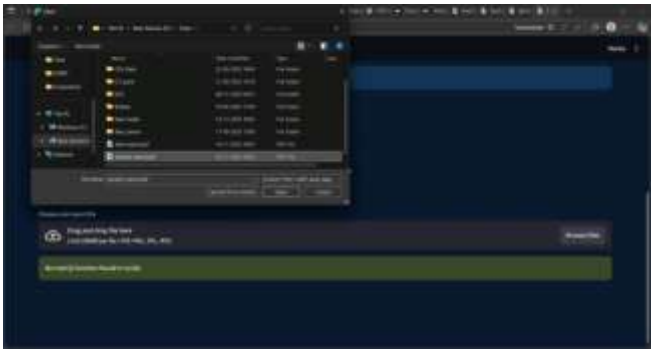

Figure 7: File Selection

The **Figure 7** shows a file upload dialog box within a health analysis system's Blood Test Module. The user is selecting a PDF file named "sample_report.pdf" from a directory on the D: drive, which contains various folders related to health data such as CT scans, ECG, kidney, and skin cancer. The upload interface allows users to drag and drop files or manually browse and select files with accepted formats including PDF, PNG, JPG, and JPEG, with a maximum size limit of 200MB. Below the file selection area, there is a message stating "No main() function found in script," indicating a possible issue or feedback related to a script execution. The background interface for the Blood Test Module maintains a dark theme consistent with earlier scenes seen in the system.

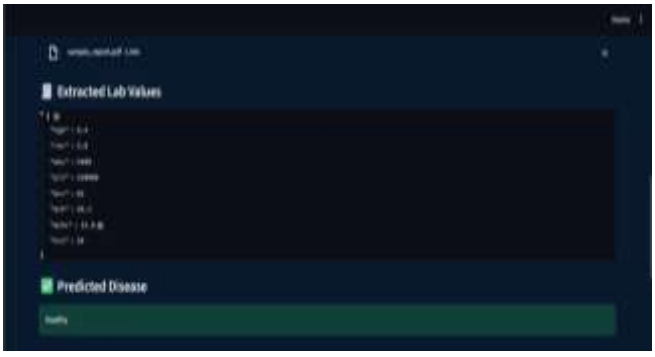

Figure 8: Predicting Process

The **Figure 8** displays the results screen of the Blood Test Module from a health analysis system. It shows the extracted lab values from a file named "sample_report.pdf," presenting key hematological metrics such as hemoglobin (hgb), red blood cell count (rbc), white blood cell count (wbc), platelets (plt), and others in a structured JSON format. Beneath the extracted data, the system provides a prediction of the patient's health status, which in this case is indicated as "Healthy" with a green checkmark. The interface maintains a dark theme and is designed to clearly present the lab data alongside the automated disease prediction.

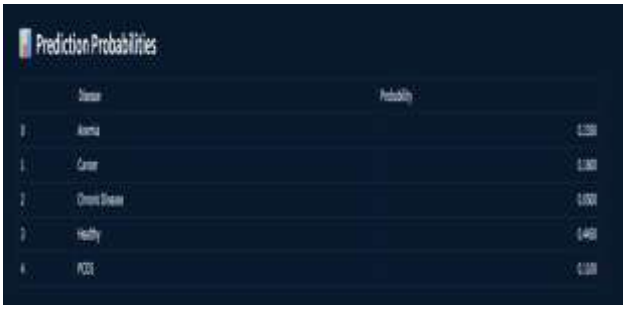

Figure 9: Prediction Probability

The **Figure 9** presents a table titled "Prediction Probabilities," which lists various diseases alongside their corresponding probability values. The diseases included are Anemia, Cancer, Chronic Disease, Healthy, and PCOS. Each disease is associated with a probability indicating the likelihood of the condition, with "Healthy" having the highest probability at 0.4450, followed by Anemia at 0.2350, Cancer at 0.1600, PCOS at 0.1100, and Chronic Disease at 0.0500. The table provides a clear overview of the predicted chances of each health status based on the analyzed data.

Figure 10: Final Health Prediction

The **Figure 10** model predicts a healthy status based on specific lab values extracted from the report, including HGB (9.4), RBC (3.8), WBC (5400), PLT (220000), MCV (82.0), MCH (26.1), MCHC (31.8), and HCT (34.0). The machine learning pipeline analyzes patterns and correlations among these clinical parameters to identify potential health conditions. For instance, elevated WBC and neutrophil counts may indicate inflammation or infection, while low hemoglobin or RBC levels can suggest anemia, and abnormal platelet counts might point to clotting issues. By considering the combination of these values, the model determines the most probable health status, providing a prediction confidence of 44.50%.

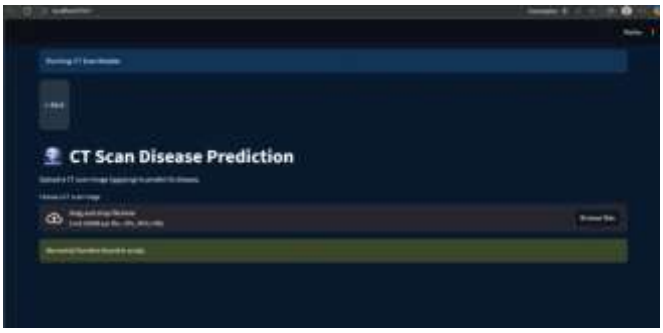

Figure 11: CT Prediction

The **Figure 11** shows the CT Scan Disease Prediction module of a health analysis system. It allows users to upload CT scan images in JPG or

PNG formats to predict potential diseases. The interface provides a drag-and-drop area and a button to browse and select files, with a file size limit of 200MB. A message displayed below the upload section indicates that no main() function was found in the script, which could imply an issue with the underlying code execution. The overall layout has a dark theme, with clear instructions and navigation options.

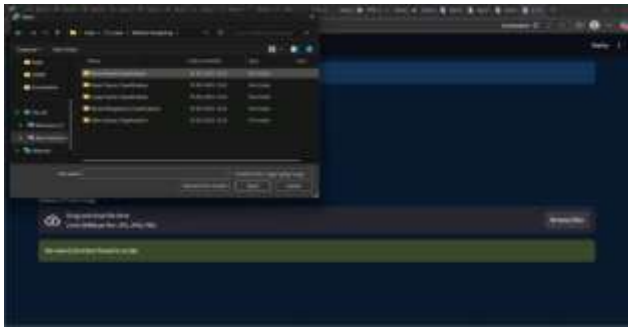

Figure 12: File Selection

The **Figure 12** shows a file selection dialog within a CT Scan Disease Prediction module of a health analysis system. The user is browsing the "Medical Imaging" folder, which contains subfolders for different classification tasks such as Bone Break Classification, Brain Tumor Classification, Lung Cancer Classification, Renal Malignancy Classification, and Skin Lesions Classification. The interface allows users to upload CT scan images in JPG, JPEG, or PNG formats with a file size limit of 200MB. A message at the bottom indicates that "No main() function found in script," suggesting a potential issue with the script execution in the system. The background interface continues the dark-themed design characteristic of the application.

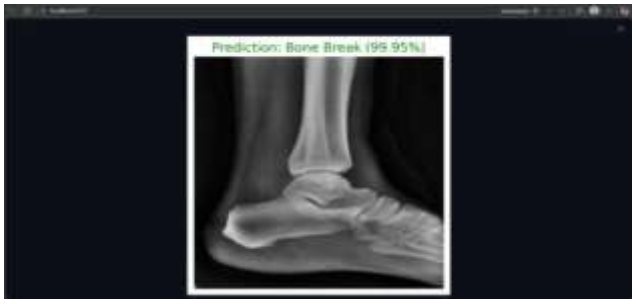

Figure 13: Bone Break Prediction

The **Figure 13** displays an X-ray of an ankle with a clear prediction result indicating a bone break. The system has identified a bone fracture with a high confidence level of 99.95%, as shown by the text above the X-ray image. The interface is simple and focused, highlighting the diagnostic outcome prominently in green text, which emphasizes the seriousness and certainty of the bone break diagnosis.

Figure 4 Final Bone Break Prediction

The **Figure 14** presents the prediction results of a CT scan analysis, indicating a bone break with a high confidence level of 99.95%. The explanation provided states that the model's prediction is based on learned spatial patterns in the scan, analyzing deep convolutional features such as edges, shapes, and textures. The confidence score reflects the model's certainty in identifying the bone fracture from the CT scan data.

Environmental and Societal Impact (Merged)

- **Digital Healthcare and Paper Reduction:** AyuSense promotes digital storage and analysis of medical data, reducing the use of paper-based medical records. This helps conserve natural resources and supports environmentally sustainable healthcare practices.
- **Reduced Travel and Carbon Emissions:** By enabling remote diagnosis and telemedicine support, the system minimizes patient travel to hospitals, leading to reduced fuel consumption, lower carbon emissions, and improved access to healthcare for rural communities.
- **Energy-Efficient and Cost-Effective Diagnosis:** The use of optimized deep learning models ensures efficient energy usage while providing accurate preliminary diagnosis. This reduces operational costs and makes healthcare more affordable for society.
- **Early Disease Detection and Preventive Care:** Early identification of health conditions such as cardiac disorders, skin diseases, pneumonia, and blood abnormalities improves patient outcomes, lowers long-term treatment costs, and reduces the burden on healthcare infrastructure.
- **Reduction in Unnecessary Medical Tests:** Accurate AI-based analysis avoids repeated or unnecessary diagnostic tests, saving medical resources, reducing energy consumption in laboratories, and minimizing patient discomfort.
- **Improved Healthcare Accessibility:** AyuSense bridges the gap between urban and rural healthcare by providing easy access to diagnostic support where specialist doctors and facilities are limited.
- **Reduced Workload on Healthcare Professionals:** Automated screening assists doctors by handling routine analysis tasks, allowing them to focus on critical cases and improving overall healthcare efficiency.
- **Ethical and Responsible Use of AI:** The system functions as a decision-support tool, ensuring transparency, data privacy, and responsible AI usage while maintaining trust between patients and healthcare providers.
- **Long-Term Sustainable Healthcare Development:** By combining environmental conservation with social well-being, AyuSense contributes to sustainable digital healthcare transformation and improved public health outcomes.

6. CONCLUSION

The AyuSense project has successfully developed an intelligent health analysis system capable of processing and interpreting diverse medical data types, including blood reports, ECG signals, skin images, and medical scans. By employing machine learning and deep learning algorithms specifically suited for each type of dataset, the system can generate accurate predictions and health insights. The integration of these models into a unified platform with a user-friendly dashboard enables users to visualize results through graphs, alerts, and detailed reports,

facilitating proactive health monitoring and informed decision-making. This approach demonstrates the potential of AI-driven solutions to enhance early detection, personalized care, and continuous health surveillance, marking a significant advancement in digital healthcare.

Looking forward, the system can be further enhanced by integrating it with hospital networks. Users could upload their medical data and receive instant predictions, while the system would suggest nearby hospitals for consultation. Moreover, the analysis reports could be directly shared with healthcare providers, and appointment scheduling could be automated to streamline further treatment. This future integration would not only improve accessibility and convenience for patients but also ensure continuity of care, enabling timely medical interventions.

By expanding real-time capabilities, data security, and cross-platform accessibility, AyuSense has the potential to become a comprehensive, scalable, and inclusive health monitoring solution, bridging the gap between patients and healthcare providers.

REFERENCES

[1] Aquil, A.; Saeed, F.; Baowidan, S.; Ali, A. M.; Elmitwally, N. S. Early Detection of Skin Diseases Across Diverse Skin Tones Using Hybrid Machine Learning and Deep Learning Models. Information 2025, 16, 152. https://doi.org/10.3390/info16020152

[2] Chen, M., Zhou, P., Wu, D., Hu, L., Hassan, M. M., & Alamri, A. (2019). AI-Skin : Skin Disease Recognition based on Self-learning and Wide Data Collection through a Closed Loop Framework. arXiv preprint arXiv:1906.01895v1 [cs.CV]. https://arxiv.org/abs/1906.01895

[3] Noor, M. N., Haneef, F., Ashraf, I., & Masud, M. (2025). Enhanced Skin Disease Classification via Dataset Refinement and Attention-Based Vision Approach. Bioengineering, 12(3), 275. https://doi.org/10.3390/bioengineering12030275

4] Makhmudov, F., Ravshanov, N., Akhmedov, D., Pekos, O., Turimov, D., & Cho, Y.-I. (2025). A Multitask Deep Learning Model for Predicting Myocardial Infarction Complications. Bioengineering, 12(5), 520. https://doi.org/10.3390/bioengineering12050520

[5] Golande, A. L., & Pavankumar, T. (2023). Optical electrocardiogram based heart disease prediction using hybrid deep learning. Journal of Big Data, 10(1), 139. https://doi.org/10.1186/s40537-023-00820-6

[6] Sharma, S., & Guleria, K. (2023). A Deep Learning based model for the Detection of Pneumonia from Chest X-Ray Images using VGG-16 and Neural Networks. Procedia Computer Science, 218, 357–366. https://doi.org/10.1016/j.procs.2023.01.018

[7] Bharati, S., Podder, P., & Mondal, M. R. H. (2020). Hybrid deep learning for detecting lung

diseases from X-ray images. Informatics in Medicine Unlocked, 20, 100391. https://doi.org/10.1016/j.imu.2020.100391

[8] Zhang M, Ye Z, Yuan E, Lv X, Zhang Y, Tan Y, Xia C, Tang J, Huang J, Li Z. Imaging-based deep learning in kidney diseases: recent progress and future prospects. Insights Imaging. 2024;15:50. https://doi.org/10.1186/s13244-024-01636-5.

[9] Reghunandanan K, Lakshmi VS, Raj R, Viswanath K, Davis C, Chandramohanadas R. A Convolutional Neural Network-Based Deep Learning To Detect Reticulocytes From Human Peripheral Blood. Intelligence-Based Medicine. 2024;10:100175.https://doi.org/10.1016/j.ibmed.2024.100175.

[10] Chattopadhyay A, Maitra M. MRI-based brain tumour image detection using CNN based deep learning method. Neuroscience Informatics. 2022;2:100060. https://doi.org/10.1016/j.neuri.2022.100060.

[11] Lv Z, Yu Z, Xie S, Alamri A. Deep Learning- based Smart Predictive Evaluation for Interactive Multimedia-enabled Smart Healthcare. ACM Trans Multimedia Comput Commun Appl. 2022;18(1s):43. https://doi.org/10.1145/3468506.

[12] Chen S. Age-appropriate design of smart senior care product APP interface based on deep learning. Heliyon. 2024;10:e28567. https://doi.org/10.1016/j.heliyon.2024.e28567.

[13] Abdul-Jabbar SS, Farhan AK. Complete Blood Count (CBC) Test. https://www.kaggle.com/datasets/ahmedelsayedtaha/complete-blood-count-cbc-test

[14] Medical Imagining (CT scan, MRI, X-ray, and Microscopic Imagery) Data - Mendeley Data https://share.google/Tfa6xbhmtQXbfXhRj

[15] Skin Disease Classification [Image Dataset] https://www.kaggle.com/datasets/riyaelizashaju/skin-disease-classification-image-dataset

[16] Chest X-Ray images (Kaggle). https://www.kaggle.com/datasets/paultimothymooney/chest-xray-pneumonia

[17] Chest X-ray (Covid-19 & Pneumonia): This dataset consists of 6,432 radiographs. It is used for multi-class classification with three categories: Pneumonia, Covid-19, and Normal.Link:https://www.kaggle.com/datasets/prashant268/chest-xray-covid19-pneumonia

[18] PTB-XL, a large publicly available electrocardiography dataset v1.0.3 https://share.google/Ot9J1AXVDLeTuFky4