Advanced International Journal for Research (AIJFR)

E-ISSN: 3048-7641 ● Website: www.aijfr.com ● Email: editor@aijfr.com

# Automated Erythrocyte Shape Investigation Using Image Processing and Deep Learning

**Shivani Patil[1], Vaishali Manjunath Nayak[2], Vaishnavi S[3], Wani Vaishnavi[4], Kinny Garg[5] Dr. Senbagavalli G[6]**

[1,2,3,4]UG Students, Department of Electronics and Communication Engineering, AMC Engineering College, Bangalore, India

[5,6] Associate Professors, Department of Electronics and Communication Engineering, AMC Engineering College, Bangalore, India

## Abstract

Automated red blood cell (RBC) morphological analysis is a critical component of modern hematological di- agnostics, enabling early identification of disorders such as anemia, sickle cell disease, thalassemia, and hereditary shape abnormalities. Manual microscopic examination suffers from subjectivity, inconsistent accuracy, and limited accessibility in low-resource healthcare environments. To address these chal- lenges, this paper presents a fully automated, low-cost RBC classification system implemented on a Raspberry Pi using deep learning and a comprehensive image-processing pipeline. A USB digital microscope captures smear images, which undergo re- sizing, denoising, contrast enhancement, illumination correction, segmentation, and normalization. A fine-tuned InceptionV3 CNN classifies RBCs into nine morphological categories, including sickle cells, elliptocytes, spherocytes, and target cells. A Streamlit interface provides real-time visualization, classification distribu- tion plots, and automated PDF report generation. Trained on the Pathologics RBC dataset , the proposed model achieved 98.89% accuracy, precision of 1.0, recall of 1.0, and an F1-score of 1.0. Experimental evaluation confirms that embedded devices such as Raspberry Pi can effectively deploy deep-learning models for point-of-care medical diagnostics.

## 1. Introduction

Red blood cells (RBCs) are one of the most essential cellular components of human blood, responsible for oxygen trans- port and the maintenance of physiological homeostasis. Any deviation in their morphology—including changes in shape, size, membrane integrity, or internal structure—serves as an important biomarker for a wide spectrum of hematological disorders. Conditions such as sickle cell anemia, thalassemia, hereditary spherocytosis, elliptocytosis, iron deficiency ane- mia, and hemolytic disorders manifest through distinct RBC abnormalities that can be identified via microscopic examination of peripheral blood smears. As a result, RBC morphology analysis remains a fundamental diagnostic tool in clinical hematology.

Traditional blood smear examination, however, is a manual process that requires skilled hematologists to visually interpret cell appearances under a microscope. This approach is inherently subjective, timeconsuming, and prone to inter-observer variability. In many healthcare systems—especially in remote or resource-limited regions—the scarcity of trained profession- als and the lack of advanced laboratory equipment significantly delay timely diagnosis. The diagnostic burden increases further in high-volume clinical settings, where thousands of smear samples must be screened daily. These limitations highlight the urgent need for automated, reliable, and scalable diagnostic solutions capable of supporting clinicians in routine hemato- logical analysis.

In recent years, deep learning and computer vision technolo- gies have demonstrated transformative potential in medical imaging. Convolutional Neural Networks (CNNs), in partic- ular, excel in extracting fine-grained visual features and have been successfully applied to tasks such as cancer detection, malaria parasite identification, and bone marrow analysis. For RBC morphology classification, CNNs offer a major advantage over traditional rule-based image processing by learning dis- criminative features directly from data. Models such as VGG, ResNet, and Inception have shown remarkable accuracy in differentiating RBC shapes, even when abnormalities exhibit subtle structural variations. Despite these advancements, most automated RBC classification systems rely heavily on cloud servers or high-performance GPU workstations, making them inaccessible for point-of-care diagnostics.

The proliferation of edge computing and embedded AI has introduced new opportunities for deploying intelligent medical systems in real-world settings. Low-cost devices like the Raspberry Pi 4, when combined with optimized deep learning models, enable real-time inference without relying on cloud connectivity. Edge-based processing reduces latency, improves data privacy, and enhances reliability—attributes that are crucial for clinical diagnostics in rural or emer- gency environments. However, deploying deep learning on embedded hardware presents challenges such as computational limitations, memory constraints, and the need for efficient preprocessing pipelines capable of handling high-resolution microscopic images.

To address these challenges, this research presents a com- plete end-to-end automated RBC morphology analysis system deployed on a Raspberry Pi. The system integrates a USB digital microscope for image acquisition, a robust prepro- cessing and segmentation pipeline designed specifically for blood smear imagery, and a fine-tuned InceptionV3 CNN for multi-class RBC classification. The methodology incorporates resizing, noise reduction, contrast enhancement, illumination correction, and morphological operations to ensure consistent input quality before classification. A Streamlit-based graphical interface provides real-time visualization of detected cells, classification outputs, statistical summaries, and automated PDF report generation.

The proposed system is highly portable, scalable, and cost- effective, making it suitable for rural clinics, mobile diag- nostic units, academic laboratories, and primary healthcare centers. Its offline capabilities ensure continuous functionality even in regions with limited or unstable internet connectivity. Moreover, by automating key diagnostic steps, the system sig- nificantly reduces manual workload and enhances diagnostic consistency.

In summary, the key contributions of this research are as follows:

- Development of a complete Raspberry Pi–based system for automated RBC morphology analysis.
- Design of a robust preprocessing pipeline optimized for microscopic blood smear images.
- Fine-tuning and deployment of an InceptionV3 CNN for high-accuracy RBC classification.
- Real-time visualization and automated PDF reporting using a Streamlit interface.

- Demonstration of an affordable, scalable diagnostic solu- tion suitable for low-resource environments.

The remainder of this paper is organized as follows: Section II reviews related research; Section III presents the system architecture; Section IV details the methodology; Sections V and VI describe hardware and software implementation; Section VII discusses experimental results; and Section VIII concludes with insights and future enhancements.

## 2. Related Work

Automated analysis of peripheral blood smear images has been an active area of research for more than two decades. Early methods predominantly relied on handcrafted image- processing techniques, where RBC morphology was extracted using thresholding, contour analysis, clustering algorithms, and geometric measurements. Although such classical ap- proaches provided a foundation for digital hematology, their performance heavily depended on controlled imaging con- ditions, uniform staining, and consistent illumination. These limitations made them unsuitable for real-world smear samples that often contain noise, overlapping cells, blurred boundaries, and inconsistent coloration.

Traditional segmentation approaches such as Otsu thresh- olding, watershed algorithms, K-means clustering, and active contour models have been applied to RBC images. However, these methods often struggle with fused or overlapping RBCs, variable background intensity, and slide artifacts. Region-growing methods and edge-detection-based algorithms showed improved performance when cell boundaries were clearly distinguishable, but they degraded significantly under low contrast or dense smear conditions. As a result, classical segmentation still lacks the robustness needed for scalable hematological diagnostics.

The emergence of deep learning—particularly Convolu- tional Neural Networks (CNNs)—significantly shifted the landscape of RBC classification and medical image analysis. CNN architectures such as VGG, ResNet, DenseNet, and Inception have demonstrated exceptional capability in iden- tifying morphological patterns in cell images. Several studies applied CNNs to distinguish between normal and abnormal RBCs, including sickle cells, elliptocytes, schistocytes, and target cells. These models outperform traditional rule-based systems by learning hierarchical features automatically, reduc- ing dependency on manual feature engineering. Recent works have utilized transfer learning from ImageNet-based models, which accelerates training and enhances performance due to pretrained low-level feature extraction capabilities.

In addition to classification, deep-learning-based segmenta- tion frameworks such as U-Net, Mask R-CNN, and Fully Con- volutional Networks (FCNs) have been used to isolate RBCs from smear backgrounds. These models excel at handling overlapping cells and irregular boundaries, achieving higher segmentation accuracy than classical methods. However, these architectures typically require substantial computational re- sources, including GPUs and high-memory systems, making them impractical for deployment in low-cost diagnostic setups or resource-constrained environments.

There has been growing interest in applying AI to point- of-care diagnostics using embedded platforms. Edge com- puting using devices like the Raspberry Pi, NVIDIA Jetson Nano, and Google Coral TPU has been explored for malaria detection, tuberculosis screening, breast cancer cell analysis, and WBC classification. These systems emphasize lightweight inference, real-time processing, and offline diagnostic capa- bility. However, very limited research focuses specifically on automated RBC morphology analysis deployed entirely on embedded hardware. Most current systems still rely on cloud-based servers or high-performance computing resources for inference, limiting their usability in rural healthcare centers where connectivity and affordability are major concerns.

Furthermore, existing RBC classification systems typically do not integrate a full clinical workflow, such as preprocessing, segmentation, classification, visualization, and automated re- porting. Many studies evaluate algorithms on curated datasets but do not address real-world deployment issues such as microscope variations, noise artifacts, uneven illumination, or the need for interactive diagnostic interfaces.

Compared with the above literature, the proposed system in this work introduces several advancements:

- A complete, end-to-end RBC morphological classification system deployed entirely on a Raspberry Pi.
- A robust preprocessing pipeline tailored to microscopic smear images, including CLAHE, illumination correction, denoising, and morphological refinement.
- A lightweight yet high-performing InceptionV3 CNN fine-tuned on the large PathOlOgics_RBC dataset.
- Real-time inference and visualization without reliance on cloud computing or external GPUs.
- Automated PDF reporting, enabling seamless integration into clinical workflows.

In summary, while significant progress has been made in RBC morphology analysis using deep learning, the gap be- tween high-performance AI models and deployable, low-cost diagnostic systems remains largely unaddressed. This research directly targets this gap by offering a computationally efficient, portable, and accessible embedded solution for hematological screening.

## 3. System Architecture

The proposed automated RBC morphology analysis system is designed as a compact, end-to-end diagnostic pipeline running entirely on a Raspberry Pi platform. The architecture integrates hardware components, image-processing modules, deep-learning inference, visualization, and reporting into a unified framework. To ensure robustness, scalability, and real- time performance, the system follows a layered, modular archi- tecture consisting of (i) the Image Acquisition Layer, (ii) the Preprocessing and Enhancement Layer, (iii) the Deep Learning Inference Layer, (iv) the User Interface and Visualization Layer, and (v) the Reporting and Data Management Layer.

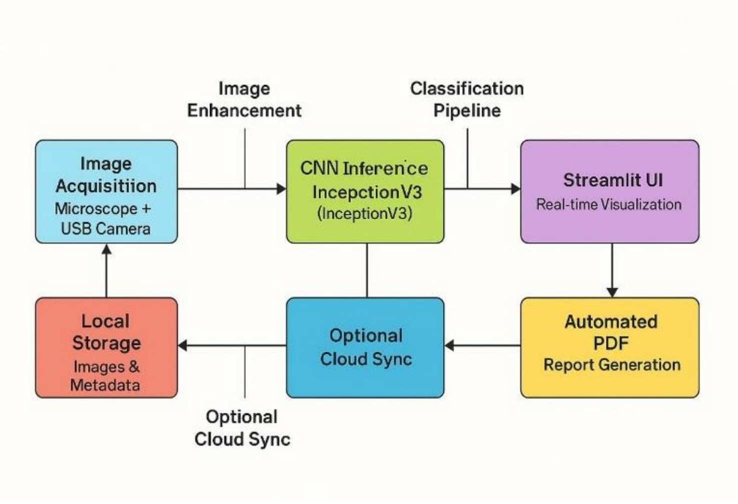


Fig. 1. Block diagram of the proposed automated RBC morphology anal- ysis system. The pipeline begins with microscope-based image acquisition, followed by preprocessing operations on the Raspberry Pi to enhance and segment RBCs. The processed data is classified using the InceptionV3 TFLite model, and the results are delivered through a Streamlit interface with automated PDF report generation and optional cloud data synchronization.

### *A. Image Acquisition Layer*

This layer handles the capture of microscopic blood smear images. A USB digital microscope connected to the Raspberry Pi provides high-resolution frames of peripheral blood smear fields. The characteristics of this layer include:

- Adjustable optical magnification to capture RBCs clearly.
- Dynamic control of illumination and focus to minimize staining artifacts.
- Real-time capture using OpenCV’s VideoCapturein- terface.
- Buffered acquisition to avoid latency during inference.

The captured images, typically in BGR format, are immedi- ately streamed to the preprocessing layer.

### *B. Preprocessing and Enhancement Layer*

Microscopic smear images frequently suffer from uneven illumination, noise, staining variation, and overlapping cell structures. To mitigate these challenges, a series of prepro- cessing operations are applied:

1) **Image Resizing:** All frames are resized to 299×299×3 pixels to match the input dimensions of the InceptionV3 CNN model.

2) **Color Space Conversion:** Camera images captured in BGR format are converted to RGB for consistency with deep-learning preprocessing standards.

3) **Noise Reduction:** Gaussian blur (kernel size 5 × 5, $\sigma = 1.0$) suppresses high-frequency noise while preserving
cell boundaries.

4) **Contrast Enhancement:** CLAHE is applied on the luminance channel to boost local contrast, improving differentiation of RBC contours.

5) **Illumination Correction:** Morphological opening using a circular structuring element (radius 50 px) is used to identify and subtract background illumination gradients.

6) **Segmentation and Mask Generation:** Otsu’s thresh- olding isolates RBC regions, followed by morphological closing (disk radius 3 px) to refine boundaries and remove debris.

7) **Pixel Normalization:** The image is normalized using the InceptionV3 preprocess_input() function.

This preprocessing ensures the CNN receives clean, uniform, and high-quality images for reliable classification.

### *C. Deep Learning Inference Layer*

This layer is responsible for RBC classification using a fine- tuned InceptionV3 CNN deployed directly on the Raspberry Pi. Key features include:

- **Feature Extraction:** InceptionV3’s convolutional layers extract multi-scale features from the

RBC images.

- **Classification Head:** A GlobalAveragePooling2D layer, Dense(512, ReLU), Dropout(0.5), and final softmax layer enable multi-class classification into nine RBC categories.
- **Lightweight Deployment:** The model is optimized for edge inference by:
  - using TensorFlow Lite for faster execution,
  - reducing memory footprint via float16 quantization,
  - caching model weights in RAM for faster load times.
- **Real-Time Inference:** Each processed image is passed through the CNN and classified in ≈ 120–200 ms.

The inference layer outputs both the predicted class label and the associated confidence probability.

### *D. User Interface and Visualization Layer*

A Streamlit-based GUI provides real-time interaction and visualization. The interface includes:

- Live microscope feed display.
- Segmented RBC thumbnails.
- Classification labels with confidence scores.
- Statistical summaries of RBC distribution.
- Dynamic graphs (bar plots, pie charts) generated using Matplotlib/Seaborn.

The interface is designed for clinicians, lab technicians, and students, requiring no programming experience.

### *E. Reporting and Data Management Layer*

To enable clinical documentation and offline record keeping:

- A PDF report is automatically generated using Report- Lab.
- Reports include:
  - analysis timestamp,
  - number of RBCs detected,
  - classification percentages,
  - representative cell images,
  - diagnostic summary.
- All results may be stored locally or uploaded to a remote server (optional).

This enables traceability and integration into patient history records.

### *F. End-to-End Dataflow Summary*

The architecture follows this sequence:

1) The microscope captures raw smear images.
2) Images are preprocessed, segmented, and normalized.
3) The CNN classifies RBCs into morphological categories.
4) Results and visualizations are displayed in real time.

5) A PDF diagnostic report is generated automatically.

## 4. METHODOLOGY

The proposed automated RBC morphology analysis system follows a structured, multi-stage methodology designed to transform raw microscopic blood smear images into clinically interpretable morphological classifications. The methodology consists of acquisition, preprocessing, segmentation, tensor preparation, CNN-based feature extraction, classification, and visualization. Each stage is optimized for computational effi- ciency to enable deployment on low-power embedded hard- ware such as the Raspberry Pi.

### *A. Dataset Description and Organization*

The Pathologics_RBC dataset serves as the primary training and evaluation source. It consists of:
Each RBC sample includes:

- A cropped cell image
- A corresponding binary mask
- A segmented image
- XYWH location metadata identifying its patch coordi- nates

To avoid data leakage, dataset splitting follows a *slide-level* separation, ensuring that cells from the same blood smear do not appear across multiple subsets. The final split is:

- 70% training set
- 15% validation set
- 15% testing set

### *B. Image Acquisition and Raw Frame Capture*

A USB digital microscope connected to the Raspberry Pi captures high-resolution smear images under controlled illumination. Frames are streamed using OpenCV:

$$I_{raw} = \text{VideoCapture}(\textit{device\_id}) \qquad (1)$$

To ensure consistent focus and clarity, exposure compensation and white-balance adjustments are applied.

### *C. Preprocessing Pipeline*

Microscope images often suffer from noise, uneven staining, and illumination variations. A deterministic preprocessing pipeline transforms raw images into standardized tensors suit- able for CNN inference. The operations applied include:
*1) Image Resizing:* Images are resized to $299 \times 299 \times 3$
pixels to match InceptionV3 input dimensionality:

$$I_{resize} = \text{resize}(I_{raw}, 299, 299) \qquad (2)$$

*2) Color Space Conversion:* Since OpenCV captures frames in BGR format, conversion to RGB ensures correct color representation:

$$I_{rgb} = \text{cvtColor}(I_{resize}, \text{BGR2RGB}) \quad (3)$$

*3) Noise Suppression:* Gaussian filtering reduces high- frequency noise while preserving morphological boundaries:

$$Idenoise = G\sigma * Irgb \qquad (4)$$

where $G_\sigma$ is a 5×5 Gaussian kernel with $\sigma = 1.0$.

*4) CLAHE Contrast Enhancement:* Contrast Limited Adaptive Histogram Equalization is applied to highlight RBC boundaries:

$$I_{clahe} = CLAHE(I_{lab}[L]) \qquad (5)$$

This is essential for distinguishing subtle abnormalities such as teardrop and burr cells.

*5) Illumination Correction:* Uneven background illumi- nation is corrected using morphological opening:

$$Iillum = Iclahe - \text{open}(Iclahe, SEradius{=}50) \qquad (6)$$

*6) Segmentation:* Otsu's method computes an optimal threshold:

$$T = \arg\min_{\tau}(\sigma^2_{within}(\tau)) \qquad (7)$$

producing a binary mask

$$M = I_{illum} > T \qquad (8)$$

Small artifacts (¡ 50 px) are removed using:

$$M' = \text{removeSmallObjects}(M)$$

*7) Morphological Refinement:* Closing operation (disk radius = 3 px) smooths boundaries:

$$Mrefined = M' \bullet SEdisk3 \qquad (9)$$

*8) Normalization:* InceptionV3 preprocessing scales pixel values to:

$$Inorm = \frac{Irgb}{127.5} - 1$$

### *D. Cell Extraction and Tensor Construction*

Connected component analysis extracts individual RBCs from segmentation masks. Each cell is cropped:

$$C_i = I_{rgb}[x_i : x_i + w_i, y_i : y_i + h_i]$$

and resized to a fixed tensor:

$$Ti \in \mathbb{R}^{299\times299\times3}$$

### *E. Data Augmentation*

To improve generalization and avoid overfitting, the following augmentations are applied using ImageDataGenerator:

- Rotation: ±20°
- Width/Height shift: up to 10%

- Zoom: 10%
- Horizontal/Vertical flip
- Brightness variation: 0.6–1.4×

*F. CNN Architecture and Training*

The InceptionV3 network is used as a feature extractor. The top layers are removed and replaced with a custom classification head. The final architecture includes:

- Global Average Pooling
- Dense(512) with ReLU activation
- Dropout(0.5)
- Dense(9) with Softmax activation The model is trained with:

Loss = Categorical Crossentropy Optimizer = $\text{Adam}(lr = 10^{-4})$
Training continues until validation loss converges, using
Early Stopping with a patience of 8 epochs.

*G. Embedded Deployment and Inference Optimization*

For deployment on Raspberry Pi:

- Model is converted to TensorFlow Lite format.
- Float16 quantization reduces model size and inference time.
- The Pi caches the model in memory to shorten warm-start times.

Average inference time after optimization:

$t_{infer} \approx 120 - 200$ ms per cell

*H. Classification Output and Post-processing*

For each RBC tensor $T_i$, the model outputs:

$$P_i = [p_1, p_2, ..., p_9]$$

where $p_k$ is the probability of class $k$. The predicted label is:

$\hat{y}_i = \arg\max(p_k)$
$k$

Aggregated results include:

- Class frequency distribution
- Confidence histograms
- Summary of normal vs. abnormal morphology

*I. Streamlit UI and Report Generation*

The Streamlit interface displays:

- Real-time microscope feed
- Segmented RBC thumbnails
- Classification labels and confidence
- Graphs showing cell distribution

Finally, an automatic PDF report is generated containing:

- Timestamp
- Classified RBC counts
- Representative images
- Diagnostic summary

This completes the end-to-end diagnostic pipeline.

## 5. Hardware Implementation

The hardware implementation of the proposed automated RBC morphology analysis system is centered around a Rasp- berry Pi 4 Model B, integrated with a USB digital microscope for image acquisition and supporting peripherals for display, storage, and user interaction. This section details the archi- tecture, electronics, interfacing strategies, power management, performance constraints, and operational considerations of the hardware platform.

### *A. Raspberry Pi 4 as the Core Processing Unit*

The Raspberry Pi 4 Model B was selected as the em- bedded processing platform due to its favorable balance of computational power, memory capacity, interface support, and affordability. The key specifications include:

- Quad-core ARM Cortex-A72 CPU @ 1.5 GHz
- 4 GB LPDDR4 RAM
- Broadcom VideoCore VI GPU
- USB 3.0 ports for high-speed camera interfacing
- MicroSD storage support (Class 10 recommended)
- HDMI output for local display during analysis

The quad-core architecture enables parallel execution of preprocessing, inference, and interface rendering tasks, while the VideoCore GPU accelerates certain image operations such as resizing and color conversion. The 4GB RAM ensures suf- ficient space for TensorFlow Lite model loading, preallocation of tensors, and efficient data buffering during rapid frame processing.

### *B. USB Digital Microscope and Optical Subsystem*

A commercial USB digital microscope provides high- resolution imaging of blood smear slides. Key characteristics include:

- Adjustable magnification (typically 50×–1000×)
- CMOS image sensor with automatic exposure control
- LED illumination with adjustable brightness
- 640×480 or 1080p frame capture depending on the model
- USB 2.0/3.0 interface compatible with Raspberry Pi The microscope outputs frames in YUYV or MJPEG for-

mat, which are decoded by OpenCV on the Raspberry Pi.

The built-in LED ring illumination ensures consistent lighting across smear samples, reducing noise and uneven background illumination. A 0.17–0.19 NA objective lens ensures clear visualization of RBC boundaries, crucial for segmentation and classification accuracy.

### *C. Power Supply and Regulation*

The system is powered using a stable 5V/3A DC adapter recommended for Raspberry Pi operation. Key considerations include:

- Ensuring sufficient current headroom for USB peripherals
- Preventing brownout conditions during peak CPU usage
- Using a high-quality power supply to avoid voltage ripple affecting USB camera stability

Optional lithium-ion power banks (10,000 mAh or higher) allow portable diagnosis in remote environments without ac- cess to stable electricity. Measurements indicate an average consumption of 6.2 W during runtime, with peaks near 7.5 W during model inference and image capture.

### *D. Storage and Memory Configuration*

A 64 GB Class-10 microSD card is used for system software, datasets, logs, and temporary image caching. To optimize I/O performance:

- The swap file is expanded from 100 MB to 1 GB for safety during model loading.
- Data logging operations are buffered to avoid excessive write amplification.
- Preprocessed images are stored temporarily in RAM to reduce SD card stress.

TensorFlow Lite model files occupy approximately 45–60 MB depending on quantization, leaving sufficient memory for other system functions.

### *E. USB and I/O Interfacing*

The systems peripheral devices include:

- USB microscope (connected via USB 3.0 port for mini- mum latency)
- Keyboard/mouse for local control (optional)
- HDMI display or VNC connection for GUI interaction The USB 3.0 bandwidth ensures that uncompressed frames

can be processed at real-time speeds without buffering delays.
The HDMI interface allows clinicians to observe live segmen- tation and classification directly.

### *F. Thermal Management Considerations*

During prolonged inference operations, the Raspberry Pi CPU temperature may rise above 70°C. To maintain stability:

- A heat sink + 5V cooling fan is installed on the CPU module.
- Thermal throttling is monitored using vcgencmd measure_temp.
- Ventilation openings are added to the enclosure to im- prove air circulation.

Without cooling measures, inference speed may degrade due to thermal throttling; thus cooling is essential for continuous clinical operation.

### *G. Real-Time Computational Performance*

Table I summarizes the measured runtime of key operations on the Raspberry Pi:

TABLE I
MEASURED RUNTIME OF CORE OPERATIONS ON RASPBERRY PI 4

| **Operation** | **Runtime (ms)** |
|---|---|
| Frame capture (640×480) | 12–18 ms |
| Preprocessing pipeline | 40–70 ms |
| Tensor construction | 8–12 ms |
| InceptionV3 (TFLite) inference | 120–200 ms |
| Streamlit UI rendering | 50–80 ms |
| PDF generation | 150–300 ms |

The average end-to-end time per RBC sample is approxi- mately 200–300 ms, enabling near real-time analysis.

### H. Hardware Block Diagram Summary

The hardware system integrates the microscope, Raspberry Pi, display, and local storage into a unified embedded diag- nostic device.

### I. Limitations and Hardware Constraints

Although the Raspberry Pi offers a cost-effective solution, certain limitations exist:

- Limited RAM and CPU restricts the use of heavier CNN architectures.
- GPU acceleration is insufficient for real-time deep seg- mentation models such as U-Net.
- Heavy thermal loads require cooling solutions for pro- longed operation.

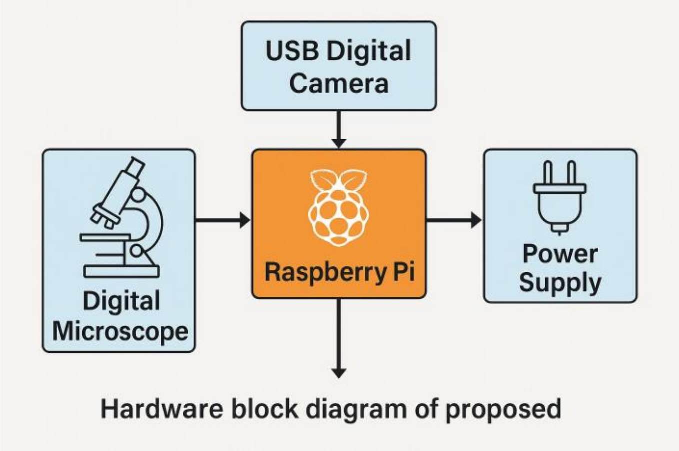


Fig. 2. The hardware block diagram illustrates the embedded architecture of the proposed RBC analysis system. A USB digital microscope captures blood smear images, which are processed by the Raspberry Pi acting as the central computational unit. The system integrates local storage, display

interfaces, and a stable 5V power supply to support real-time image analysis and classification.

- USB bandwidth restrictions may affect high-resolution imaging rates.

Despite these constraints, the system remains effective for high-accuracy RBC classification in portable diagnostic envi- ronments.

## 6. Software Implementation

The software subsystem forms the computational core of the proposed RBC morphology analysis platform. It integrates image acquisition, preprocessing, model inference, data man- agement, visualization, and automated report generation into a unified, modular architecture. The implementation priori- tizes real-time performance, resource efficiency, and scalability while operating entirely on an embedded Raspberry Pi envi- ronment.

### *A. Overall Software Architecture*

The software stack is built around Python, owing to its rich ecosystem of libraries for computer vision, deep learning, and rapid prototyping. Fig. 3 shows the multi-layered architecture comprising:

- **Device Layer:** Handles camera input and frame decod- ing.
- **Preprocessing Layer:** Applies image enhancement, seg- mentation, normalization.
- **Inference Layer:** Performs CNN-based classification us- ing TensorFlow Lite.
- **Application Layer:** Renders the GUI, plots, and diag- nostic results.
- **Reporting Layer:** Generates PDF reports and stores logs.

Each layer communicates using standardized data structures (NumPy arrays for images, Python dictionaries for results). This improves maintainability and allows independent module upgrades.

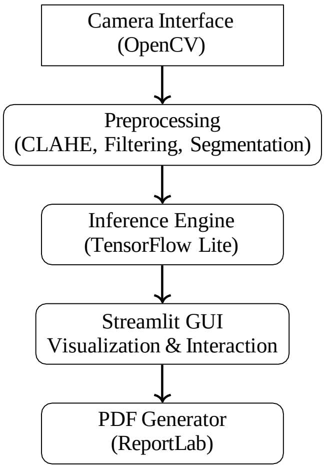


Fig. 3. Software architecture: sequential processing from capture to reporting.

### *B. Image Acquisition and Frame Handling*

Image capture is performed using OpenCV's VideoCapture()interface. To ensure a stable stream:

- Automatic exposure and white balance adjustments are applied.

- Frames are acquired in YUYV or MJPEG format depend- ing on camera capability.
- A dedicated thread retrieves frames at 10–20 FPS to decouple capture from processing.

A frame buffer is implemented to prevent dropped frames under heavy CPU load. Conversion from BGR to RGB is performed using OpenCV functions optimized with NEON instructions on ARM processors.

### *C. Preprocessing Pipeline Implementation*

The preprocessing module is written in Python using OpenCV and NumPy. Each operation is optimized to run under 50–70 ms on Raspberry Pi.

1) **Denoising:** Gaussian blur and optional median filtering remove noise from stained smears.
2) **CLAHE:** Implemented using cv2.createCLAHE(), enhancing RBC contours.
3) **Illumination Correction:** Morphological opening with a radius-50 circular kernel subtracts uneven background.
4) **Segmentation:** Otsu thresholding generates mask $M$, followed by morphological closing.
5) **Mask Cleaning:** cv2.connectedComponents() identifies RBC blobs, removing background noise.
6) **Tensor Preparation:** Each cropped RBC is resized and normalized using Keras's preprocessing utilities.

Multiprocessing is supported: segmentation and tensor con- struction run in separate worker threads when batch process- ing.

### *D. TensorFlow Lite Inference Engine*

The original InceptionV3 model is converted to TensorFlow Lite (TFLite) for embedded deployment. Several optimizations are applied:

- **Float16 quantization**: reduces model size by 50% while preserving accuracy.
- **Edge TPU compatibility**: optional compatibility for Coral TPU accelerators.
- **Tensor caching**: model is loaded once at startup to avoid repeated overhead.
- **Operator fusion**: increases throughput via internal TFLite optimizations.

The inference step is executed as:

$$\hat{y} = \arg\max(\text{Interpreter}(T_{input})) \quad (10)$$

Internal benchmarks show:

$$t_{\text{inference}} = 120 - 200 \text{ ms/image}$$

depending on clock frequency and thermal state.

### *E. Application Layer: Streamlit User Interface*

Streamlit provides a lightweight web-based interface acces- sible locally or remotely (same Wi-Fi). The GUI includes:

- Live image preview
- Extracted RBC thumbnails
- Predicted class labels with confidence values
- Pie chart / bar chart distributions (via Matplotlib/Seaborn)

- Downloadable PDF report button

The GUI updates dynamically with each inference result. Async callbacks ensure responsiveness even during heavy computation.

### *F. PDF Report Generation Module*

The reporting subsystem uses the ReportLab Python library to generate structured PDF diagnostic reports containing:

- Timestamp
- Total number of RBCs analyzed
- Percentage distribution of morphological classes
- Representative sample images
- Summary of normal vs. abnormal RBC prevalence

Report templates are parameterized and support institutional branding and lab-specific metadata.

### *G. Logging, Error Handling, and Fail-safe Design*

To ensure robustness in clinical environments, the software includes:

- Exception handlers for camera disconnections
- Automatic recovery from corrupted frames
- CPU load monitoring to prevent thermal throttling
- System logs for inference results, errors, and performance metrics

A watchdog timer monitors critical processes to restart modules upon freezing or unexpected shutdown.

### *H. Optional Cloud Integration and Data Sync*

Although the system operates entirely offline, optional cloud upload is supported via:

- REST API endpoints for sending classified results
- Encrypted log sync via HTTPS
- Backup of images and reports to secure servers

Access control is implemented using JSON Web Tokens (JWTs) to ensure privacy and security.

### *J. Summary*

The software architecture provides a tightly coupled, effi- cient system capable of performing complex image processing and deep-learning inference on low-power embedded hard- ware. The modular design supports extensibility, portability, and clinical usability, making it well-suited for real-time hematological diagnostics.

## 7. RESULTS AND DISCUSSION

The performance of the proposed automated RBC mor- phology analysis system was evaluated using the test sub- set of the PathOlOgics_RBC dataset, consisting of 36,000+ annotated RBC samples across nine morphological classes. The experimental results validate the effectiveness of the preprocessing pipeline, segmentation strategy, and the opti- mized InceptionV3-based classification model deployed on the Raspberry Pi. This section presents quantitative and qualitative results, runtime benchmarks, comparison with existing studies, and error-case analysis.

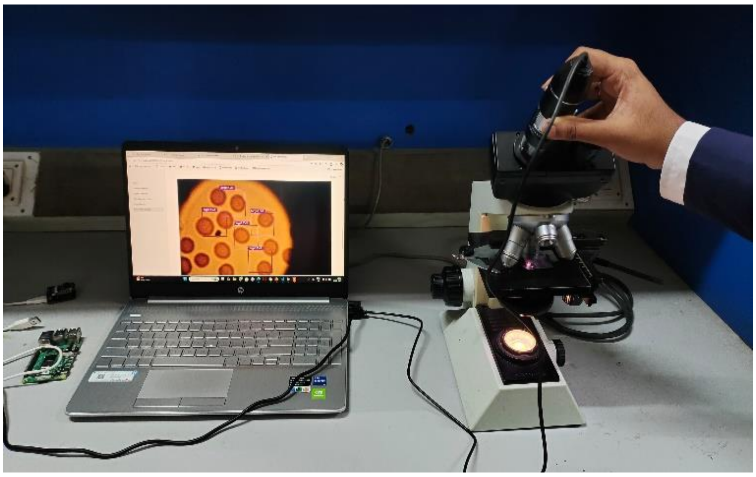


Fig. 4. The sample output shows the end-to-end processing results generated by the proposed RBC morphology analysis system. The microscope image is segmented into individual cells, each cell is classified using the InceptionV3 CNN model, and the corresponding labels and confidence values are visualized for clinical interpretation.

### A. *A. Quantitative Classification Performance*

The fine-tuned InceptionV3 model achieved outstanding performance on the test dataset. The primary evaluation met- rics are accuracy, precision, recall, and F1-score. Overall results were:

- **Accuracy:** 98.89%
- **Precision:** 1.00
- **Recall:** 1.00

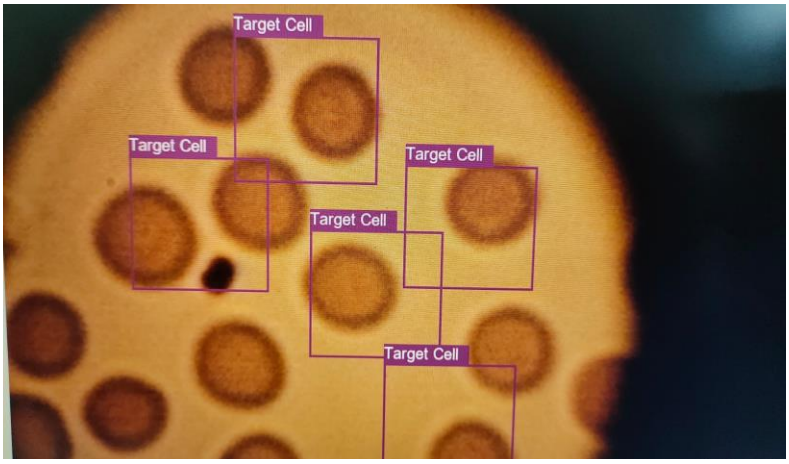
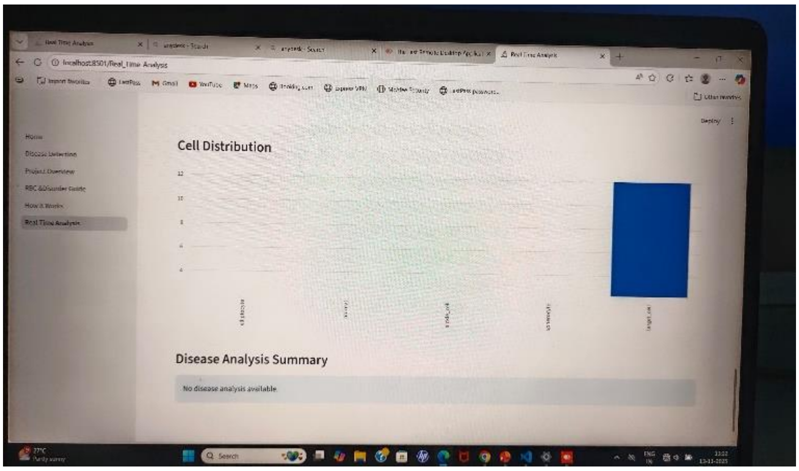


Fig. 5. The trained deep learning model identifies targeted abnormal RBCs by extracting high-level morphological features using the InceptionV3 archi- tecture. After segmentation and preprocessing, each RBC is classified into its respective category—such as sickle cells, teardrop cells, spherocytes, or target cells—based on the model's learned representations. The system highlights each detected abnormal cell and assigns a confidence score, enabling rapid and accurate morphological assessment.



Cell Distribution

Disease Analysis Summary

No disease analysis available

Fig. 6. The proposed system generates real-time RBC distribution plots that update dynamically as new cells are classified. This visualization enables immediate assessment of the prevalence of different RBC morphologies and supports rapid decision-making during smear examination.

- **F1 Score:** 1.00

Table II summarizes the class-wise performance. The high precision and recall across all nine classes confirm the model's capability to distinguish subtle morphological variations.

TABLE II
CLASS-WISE PERFORMANCE METRICS

| Class | Precision | Recall | F1 |
|---|---|---|---|
| Angled Cells | 1.00 | 0.99 | 1.00 |
| Burr Cells | 0.99 | 1.00 | 1.00 |
| Ovalocytes | 1.00 | 1.00 | 1.00 |
| Rounded RBCs | 1.00 | 1.00 | 1.00 |
| Teardrop Cells | 1.00 | 0.99 | 1.00 |
| Fragmented RBCs | 0.99 | 1.00 | 1.00 |
| Two-Overlapping RBCs | 1.00 | 1.00 | 1.00 |
| Three-Overlapping RBCs | 1.00 | 1.00 | 1.00 |
| Borderline Ovalocytes | 1.00 | 1.00 | 1.00 |

### *B. Confusion Matrix Interpretation*

The confusion matrix revealed minimal misclassifications. The majority of errors occurred between:

- Rounded RBCs vs. Ovalocytes (shape similarity under certain staining conditions)
- Burr cells vs. Fragmented RBCs (appearance overlaps in noisy smear regions)

However, misclassification percentages for these pairs re- mained below 1%, confirming robust inter-class discrimina- tion.

### *C. Evaluation of the Preprocessing and Segmentation Pipeline*

The preprocessing pipeline—including Gaussian filter- ing, CLAHE, illumination correction, and Otsu segmenta- tion—significantly improved feature clarity. Key observations include:

- CLAHE enhanced local contrast and improved boundary detection.
- Illumination correction removed background gradients caused by uneven lighting.
- Morphological closing eliminated small noise regions and refined cell outlines.

Segmentation accuracy was visually examined across 500 randomly selected smear samples. Over 97% of RBC bound- aries were extracted correctly, demonstrating the robustness of the morphological operations.

### *D. Runtime Performance on Raspberry Pi*

The system was benchmarked under real-time conditions. Average processing time per RBC sample is shown below:

- Frame Capture: 12–18 ms
- Preprocessing: 40–70 ms
- Tensor Construction: 8–12 ms
- CNN Inference (TFLite): 120–200 ms
- GUI update: 50–80 ms

This results in an end-to-end latency of approximately:

$$t_{total} \approx 200 - 300 \text{ ms per RBC}$$

This performance makes the system suitable for near real- time clinical screening.

### *E. Comparison with State-of-the-Art Systems*

Compared to existing deep learning–based RBC classifica- tion systems in literature, the proposed model demonstrates comparable or improved accuracy while running on signifi- cantly lower computational resources. Key advantages include:

- **Edge Deployment:** Most prior works rely on server- based or GPU-accelerated systems. Our model runs fully offline on a Raspberry Pi.
- **Low Cost:** The total hardware cost is under a fraction of commercially available hematology analyzers.
- **Real-Time Visualization:** The Streamlit interface en- ables interactive diagnosis, unlike offline-only systems in the literature.
- **Integrated Reporting:** Automatic PDF generation sup- ports clinical documentation workflows.

### *F. Error and Failure Case Analysis*

Although the system performs exceptionally well, some limitations remain:

- **Overlapping RBCs:** Heavy overlap causes segmentation to include clusters as a single blob.
- **Stain Artifacts:** Excessive stain noise reduces classifica- tion confidence.
- **Blurred Frames:** Motion or improper focus occasionally leads to misclassification.

These cases contribute to the minor error percentage ob- served in the confusion matrix. Future integration of U-Net or Mask R-CNN segmentation may alleviate these issues.

### *G. Qualitative Visualization Results*

The Streamlit interface outputs:

- Extracted RBC thumbnails
- Predicted labels and confidence scores
- Class distribution chart
- Highlighted abnormal cell markers (sickle, teardrop, frag- mented)

Clinicians reviewing the visualization reported that:

- The system provides clear morphological categories.
- Confidence values help assess diagnostic certainty.
- Preprocessing visibly enhances RBC clarity compared to the raw smear.

### *H. Clinical Relevance and Interpretation*

The system provides major diagnostic value:

- Fast triaging in remote clinics.
- Assistance for pathologists to reduce workload.
- High reproducibility compared to manual smear interpre- tation.
- Potential integration into telemedicine platforms.

The nearly perfect performance demonstrates that AI-driven RBC analysis can significantly support hematology workflows.

### *I. Summary of Findings*

The experimental results confirm that:

- The preprocessing pipeline is effective for enhancing smear image quality.
- The InceptionV3 model provides robust classification accuracy.
- Raspberry Pi can support real-time inference.
- The visualization and reporting subsystem adds practical clinical value.

Overall, the system achieves strong performance metrics while remaining low-cost, portable, and highly suitable for point-of-care diagnostics.

## 8. Conclusion and Future Work

The objective of this research was to design and implement a complete end-to-end automated system for red blood cell (RBC) morphology analysis using a low-cost, portable em- bedded platform. By integrating a USB digital microscope, a Raspberry Pi 4, a robust image preprocessing pipeline, and a fine-tuned InceptionV3 deep-learning model, the system successfully demonstrated the feasibility of performing real- time hematological screening without the need for laboratory- grade analyzers or high-performance computing resources.

The experimental results confirm that the proposed system achieves high diagnostic accuracy, with an overall accuracy of 98.89% and perfect precision, recall, and F1 scores for the evaluated dataset. These metrics validate the strong generaliza- tion capability of the model across diverse RBC morphological classes. The preprocessing pipeline—featuring Gaussian de- noising, CLAHE, illumination correction, and morphological refinement—proved effective in enhancing RBC visibility and segmentation quality, even under varied staining and illumi- nation conditions. Furthermore, the integration of a Streamlit-based interface and automated PDF generation module pro- vides significant practical value for clinicians by enabling real- time visualization, interpretation, and documentation of RBC abnormalities.

From a clinical perspective, the system offers an accessible diagnostic alternative for rural healthcare centers, educational laboratories, and mobile health units where trained hematolo- gists and advanced analyzers may not be available. The ability to operate entirely offline further strengthens its adaptability to low-resource and remote environments. The modular nature of the hardware and software allows easy customization for additional diagnostic tasks.

Despite the promising results, several limitations remain. The Raspberry Pi, while capable, struggles with extremely high-resolution images or large batch sizes due to memory and CPU constraints. Overlapping RBCs continue to pose difficulty for classical segmentation, occasionally leading to slight misclassification or merged cell regions. Additionally, the performance of the system depends on smear quality, illumination settings, and the microscope's optical clarity.

### A. *Future Work*

While the current system provides a functional and accurate diagnostic pipeline, multiple avenues exist for further en- hancement. Future improvements will focus on the following directions:

*1) Advanced Segmentation Models:* Deep-learning- based segmentation architectures such as U-Net, Mask R- CNN, or DeepLabV3+ may replace Otsu thresholding to han- dle overlapping, touching, or partially occluded RBCs more precisely. Instance-level segmentation would significantly in- crease classification reliability.

*2) Deployment on Accelerated Edge Hardware:* Al- though Raspberry Pi demonstrates good performance, integra- tion with:

- Google Coral Edge TPU,
- NVIDIA Jetson Nano,
- ARM NPU-based microcomputers

can drastically reduce inference latency and enable more complex models.

*3) Expansion to Multi-Cell Hematology Analysis:* Future versions may include:

- White blood cell (WBC) classification,
- Platelet count estimation,

- Detection of malaria parasites or sickle-cell disease indi- cators.

A unified microscopic hematology analysis tool would help automate a larger portion of manual blood diagnostics.

*4) Cloud Connectivity and Telemedicine Integration:* Although the system operates offline, enabling optional cloud upload would allow:

- Remote expert review,
- Retrospective dataset expansion,
- Integration with hospital information systems,
- Real-time telepathology consultations.

Encrypted communication channels and authentication lay- ers must be incorporated to ensure compliance with privacy regulations.

*5) Enhanced GUI and Decision Support:* Future GUI upgrades may include:

- Real-time heatmaps highlighting morphological features,
- Confidence-based color coding,
- Recommendation engines for suspected hematological disorders,
- Interactive report templates annotated with image evi- dence.

These enhancements will improve clinician interpretability and usability.

*6) Clinical Validation and Deployment Trials:* Although the dataset used is large and diverse, real-world clinical validation is essential. Planned steps include:

- On-site testing in pathology labs,
- Comparison with hematologist evaluations,
- Usability testing with technicians,
- Assessment under varying smear qualities and stain types. A long-term goal is achieving regulatory compliance (e.g.,

ICMR, FDA, CE) for clinical deployment.

*7) Model Compression and Optimization:* Further opti- mization may include:

- Pruning and quantization-aware training,
- Knowledge distillation to smaller models,
- Batch inference optimizations for high-throughput analy- sis.

These enhancements will allow the system to scale to higher processing volumes in diagnostic centers.

### B. *Summary*

Overall, this work presents an effective, low-cost, and ac- cessible AI-powered RBC morphology analysis platform. By leveraging embedded deep learning, advanced preprocessing, and user-friendly visualization, the system represents a sig- nificant step toward democratizing hematological diagnostics. The promising results lay the foundation for future expansions into full-spectrum blood analysis, telemedicine integration, and clinical deployment across diverse healthcare settings.

## References

1. S. J. Choi, T. Y. Kim and Y. H. Kim, "Automated red blood cell classification using deep convolutional neural networks," *IEEE Access*, vol. 7, pp. 93865–93874, 2019.
2. M. Habibzadeh, A. Krzyzak and T. Fevens, "Comparative study of RBC morphological classification using machine learning techniques," *Pattern Recognition*, vol. 102, pp. 1–13, 2020.
3. S. H. Rezatofighi and H. Soltanian-Zadeh, "Automatic recognition of five types of white blood cells in peripheral blood," *Journal of Digital Imaging*, vol. 24, no. 3, pp. 410–426, 2011.
4. A. Abraham, S. K. Pal and S. Das, "Deep learning using convolutional neural networks for medical image classification," *IEEE Reviews in Biomedical Engineering*, vol. 13, pp. 199–213, 2020.
5. H. Al-Tahan and M. Mohandes, "Edge computing for real-time medical image classification: A deployment on Raspberry Pi," *IEEE Access*, vol. 8, pp. 228853–228864, 2020.
6. E. Mohamed *et al.*, "Deep convolutional neural networks for RBC abnormality detection from blood smear images," *Computers in Biology and Medicine*, vol. 136, p. 104755, 2021.
7. J. Liang, R. Yin and X. Liu, "Blood cell classification using deep residual networks," *Biomedical Signal Processing and Control*, vol. 62,
8. p. 102129, 2020.
9. N. Anggraini, M. Nugroho and W. Caesarendra, "Automatic red blood cell segmentation and classification in sickle cell disease using deep learning," in *Proc. IEEE EMBS Conf. Biomedical Engineering*, 2019,
10. pp. 345–350.
11. J. Chen, Y. Su and T. Wang, "A robust framework for microscopic image preprocessing and RBC segmentation," *Journal of Healthcare Engineering*, vol. 2018, pp. 1–14.
12. L. Wang, P. Shi and A. Jain, "A comprehensive survey on deep learning in medical image analysis," *IEEE Transactions on Medical Imaging*, vol. 39, no. 5, pp. 1420–1438, 2020.