QUANTITATIVE ANALYSIS OF BLOOD CELL COMPONENTS AND DETECTION OF MALARIAL PARASITE (P.VIVAX) USING FASTER R-CNN

Shruthi N G, Maddi Patla Jahnavi, Maithry V Pappu,
Preksha Chandrakant Wali and Sri Lakshmi A Nair,

Department of Computer Science and Engineering, Acharya Institute of Technology, Bangalore, India

ABSTRACT

This project introduces an advanced automated system utilizing the Faster R-CNN architecture for precise detection of red blood cells (RBCs), white blood cells (WBCs), platelets, and the malarial parasite Plasmodium vivax in blood smear images. To enhance our dataset, we employ two types of Generative Adversarial Networks (GANs): one to generate new, diverse images and Real ESRGAN to improve the resolution and quality of these images, thereby increasing the robustness and performance of our system. Aimed at aiding medical professionals in diagnosing blood disorders and malaria, our system provides rapid and reliable microscopic sample analysis. Extensive experimentation confirms our method's efficacy in accurately identifying various blood components and malaria parasites, demonstrating its potential to revolutionize medical diagnostics and significantly improve patient outcomes in hematology and infectious diseases.

KEYWORDS

Faster R-CNN, blood cell detection, malaria diagnosis, Plasmodium vivax, medical image graphical analysis,GAN, Real- ESRGAN

1. INTRODUCTION

Red blood cells (RBCs), white blood cells (WBCs), and platelets constitute the fundamental components of the circulatory system, playing indispensable roles in maintaining bodily homeostasis and safeguarding against pathogens. RBCs, characterized by their distinctive biconcave shape and absence of nuclei, are laden with hemoglobin, a protein pivotal for oxygen transport. Upon binding oxygen in the lungs, RBCs facilitate its distribution to tissues throughout the body, sustaining cellular metabolism and overall vitality. Conversely, WBCs encompass a diverse array of cell types, including neutrophils, lymphocytes, monocytes, eosinophils, and basophils, each endowed with specialized functions crucial for immune surveillance and response. Neutrophils, for instance, serve as frontline defenders, engaging in phagocytosis and inflammation, while lymphocytes orchestrate adaptive immune responses, generating antibodies and coordinating cellular immunity. Additionally, platelets, diminutive cell fragments derived from megakaryocytes, play a pivotal role in hemostasis and clot formation. In the event of vascular injury, platelets aggregate at the site, forming a plug to staunch bleeding and expedite tissue repair.Accurate quantification and characterization of these blood cell populations serve as

linchpins in diagnosing hematological disorders, monitoring disease progression, and guiding therapeutic interventions.

Malaria, a vector-borne disease caused by Plasmodium parasites, remains a significant global health burden. Plasmodium vivax, known for its periodic febrile paroxysms and dormant liver stages, poses unique challenges. P. vivax exhibits a broader geographical distribution compared to other Plasmodium species, spanning tropical and temperate regions. One distinguishing feature is the formation of dormant liver stages called hypnozoites, leading to relapses months or years after the initial infection. Managing P. vivax malaria requires integrated strategies that include diagnosis, treatment of acute infections, and preventing relapses.

In addition to its diagnostic utility, the integration of Faster R-CNN for precise cell detection with Generative Adversarial Networks (GANs) for dataset augmentation presents a multifaceted approach to improving medical diagnoses. We further enhance the quality of our augmented dataset using Real ESRGAN (Enhanced SuperResolution Generative Adversarial Network), which significantly improves the resolution and clarity of generated images, contributing to more accurate and reliable analyses. By harnessing the power of these advanced technologies, we aim to enhance model performance, enable swift and accurate diagnoses, and propel advancements in malaria control. This innovative fusion of machine learning and medical science holds the promise of revolutionizing disease detection and management, ultimately leading to improved patient outcomes and global health outcomes.

2. REVIEW OF RELATED LITERATURE

[1] In sub-Saharan Africa, malaria is a deadly endemic disease exacerbated by limited expertise for accurate diagnosis, often leading to subjective results. Machine learning, particularly deep learning, offers promising solutions for medical image analysis, aiding in prompt disease control interventions. This study evaluates and compares the performance of three pre-trained deep learning architectures—faster R-CNN, SSD, and RetinaNet—on a dataset of thick blood smear images using TensorFlow object detection API. Faster R-CNN demonstrates superior performance with a mean average precision of over 0.94, while SSD emerges as the best model for mobile deployment. This research highlights the potential of deep learning in improving malaria detection accuracy and mobile healthcare solutions. CapsNet struggles with multilabel RBC classification due to overlapping cells. Faster R-CNN model improves blood cell detection accuracy. Automated image processing aids in disease prediction from microscopic blood images. Utilizes Faster R-CNN for precise blood cell detection.

Improved MAP and time performance by modifying anchor box ratio.Automated system for health anomaly detection through blood cell analysis. Potential for predicting and detecting diseases through AI-driven blood cell analysis. Simplifying disease detection for cost-effective and automated healthcare.

[2] We utilize an object detection model, Faster R-CNN, originally designed for natural images, for the first time to identify and classify cells in bright field microscopy images of malariainfected blood. This task is challenging due to variations in cell shape, density, and colour, as well as limited annotated data and imbalanced class distribution. Our dataset consists of 1300 fields of view containing approximately 100,000 individual cells. Faster R-CNN, pre-trained on ImageNet and fine-tuned with our data, outperforms a baseline method based on cell segmentation, feature extraction, and random forest classification. This study demonstrates the effectiveness of deep learning in biological image analysis, surpassing traditional approaches and approaching human performance levels. Three pre-trained deep learning models, including Faster R-CNN, SSD, and RetinaNet, were applied to a medical dataset for malaria parasite detection. Faster R-CNN excelled in accuracy with a mean average precision over 0.94, while SSD was optimal for mobile

deployment. The models were trained using the Tensorflow Object Detection API, offering implementations of state-of-the-art deep learning models like Faster R-CNN built on ResNet50 and ResNet101 architectures.

[34] Real-ESRGAN is a powerful tool for image restoration, particularly in addressing compression artifacts. It leverages generative adversarial nets to enhance creativity and pattern recognition. Super-Resolution (SR) algorithms play a crucial role in advancing spatial resolution without sensor modifications. By optimizing Real-ESRGAN with GPU tensors, processing speed is significantly improved, leading to enhanced segmentation model accuracy and efficiency.

[35]The Generative Adversarial Nets framework involves a generative model and a discriminative model in a minimax game. The discriminator is trained to differentiate between data and generated samples, while the generator aims to produce samples that fool the discriminator. The training objective maximizes loglikelihood for estimating conditional probabilities. The theoretical results show that the generator defines a probability distribution based on samples obtained, and the discriminator converges to differentiate between data and generated samples. The gradient of the discriminator guides the generator to regions likely classified as data, leading to a convergence point where both models cannot improve further.

[7] Generative Adversarial Network (GAN) with enhanced loss functions to generate a diverse and high-quality X-ray image dataset for object detection. The training involved 1038 images from the GDX-ray dataset, with adjustments made to the generator and discriminator models through back-propagation. The proposed method demonstrated a remarkable 99.83% accuracy in object detection, surpassing other image generation techniques. Faster R-CNN was employed to validate the generated X-ray dataset, showcasing improved performance in object detection results compared to existing methods.

3. METHODOLOGY

GAN

A Generative Adversarial Network (GAN) emanates in the category of Machine Learning (ML) frameworks. These networks have acquired their inspiration from Ian Goodfellow and his colleagues based on noise contrastive estimation and used loss function used in present GAN (Grnarova et al., 2019) [23].

A discriminative model that learns to identify whether a sample is from the data distribution or the model distribution opposes the generative model in the adversarial networks paradigm. Comparing the discriminative model to the police and their efforts to find counterfeit money, the generative model may be compared to a group of counterfeiters attempting to create counterfeit money and utilize it covertly. The rivalry between the two teams in this game pushes them to keep refining their techniques until the fakes can no longer be distinguished from the original items [22].

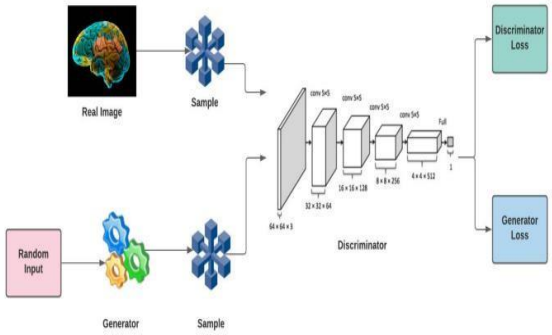


Fig 1. Block diagram of GAN[22]

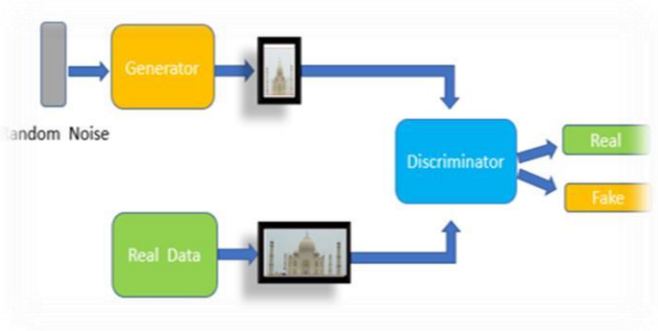


Fig 2. Simplified GAN working

Unlike the traditional model, a GAN implements two different networks and a method of combative training.GAN uses a back-propagation mechanism that does not require complex Markov chains and produces clearer and more realistic samples. It has been successfully applied to scenes such as image generation [25], video generation [26], picture style migration [27], and image completion [28]. The framework of GAN includes a pair of models: a discriminator D and generator G. The purpose of discriminator is to distinguish real training data from synthetic images to maximize the discriminant accuracy, and the generator tries to fool the discriminator. Concretely, D and G play the following game on V(D; G)

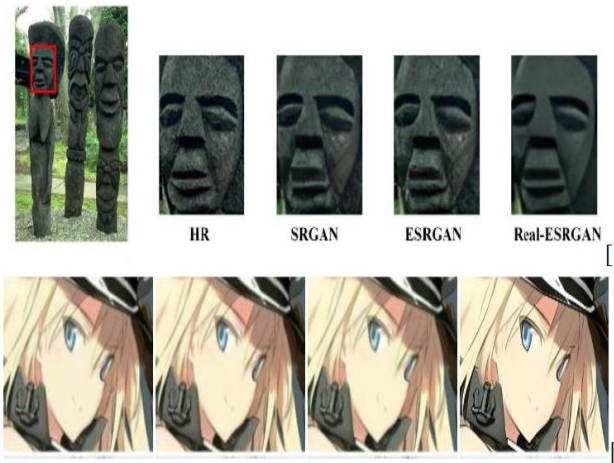
$$\min_G \max_D V(D,G) = \mathbb{E}_{x\sim p_{\text{data}}(x)}[\log D(x)] + \mathbb{E}_{x\sim p_z(z)}[\log(1-D(G(z)))] \quad [24]$$

Where, $P_{data}(x)$ stands for the real data x distribution and $P_z(z)$ stands for the model distribution. Discriminator is trained in the direction of maximizing the objective function $V_{GAN}(D, G)$, in contrast, Generator is trained to minimize $V_{GAN}(D, G)$. [7] In other words, generator and discriminator repeat adversarial competitive learning and ultimately complete a generation model that prevents the discriminator from distinguishing the data synthesized by the generator. In GANs, the competition between the generator and the discriminator is trained in the direction of solving the min-max problem and can be defined by the above equation. Through this learning process, generator G and discriminator D alternately optimize the necessary generation and discrimination networks until they reach the equilibrium point. [7]

a) Real ESRGAN

A succinct overview of SRGANs reveals the following steps: Input Low-Resolution Images: Provide low-resolution images as input to the generator. Generate Super-Resolution Images: The generator processes the input, producing high-resolution images as outputs. Discrimination Process: Subject the generated images to scrutiny by the discriminator, which evaluates their authenticity. Perceptual Enhancement: Incorporate a VGG net to introduce perceptual loss on a pixel-wise level, enhancing the sharpness of the generated synthetic images.[29]

ESRGAN thus represents a refined iteration in the evolution of super-resolution GANs (SRGANs), focusing on optimizing training efficiency and reducing complexity while maintaining the core principles of image super-resolution through adversarial learning. Enhanced Super-Resolution Generative Adversarial Networks (ESRGANs) introduce notable advancements and optimizations to the established framework of SRGANs. [29] Real-ESRGAN, an enhanced version of ESRGAN, represents a more practical solution for real-world image restoration by effectively addressing issues such as the removal of bothersome compression artifacts.[29]



[30]

[31]

Fig 3. The SR results of x4 for SRGAN, ESRGAN and Real- ESRGAN

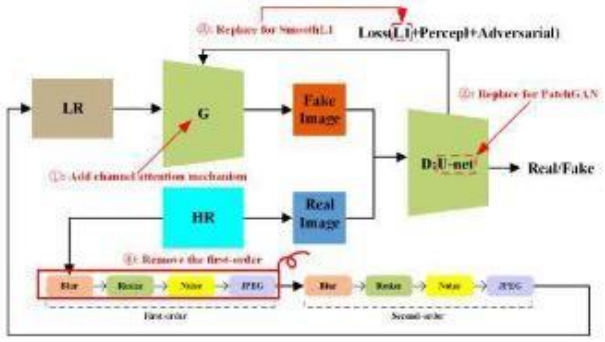


Fig 4. Real ESRGAN framework[29] Generally, the ground-truth image y is first convolved with blur kernel k. Then, a down sampling operation with scale factor r is performed. The low-resolution x is obtained by adding noise n. Finally, JPEG compression is also adopted, as it is widely-used in real-world images.[31]

$$\boldsymbol{x} = \mathcal{D}(\boldsymbol{y}) = [(\boldsymbol{y} \circledast \boldsymbol{k}) \downarrow_r + \boldsymbol{n}]_{\text{JPEG}}, \quad [31]$$

Blur

We typically model blur degradation as a convolution with a linear blur filter (kernel). Isotropic and anisotropic Gaussian filters are common choices. Fora Gaussian blur kernel k with a kernel size of 2t + 1,its (i, j) ∈ [−t, t] element is sampled from a Gaussian distribution, formally [31]

$$\boldsymbol{k}(i,j) = \frac{1}{N}\exp(-\frac{1}{2}\boldsymbol{C}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{C}), \quad \boldsymbol{C} = [i,j]^T, \quad [31]$$

Resize (Down sampling)

For synthesizing low resolution images essential in SR applications, downsampling methods like area, bilinear andbicubic interpolation are used, each producing different effects, from blurring to potential overshoot artifacts.

Noise

Noise in images can vary, with common types including color noise, where RGB channels are independently affected, and grey noise affecting all channels uniformly. Poisson noise, another type is based on the Poisson distribution and relates to variations in photon numbers at given exposure levels, typically modeling sensor noise.

JPEG Compression

JPEG compression is a commonly used technique of lossy compression for digital images. It first converts images into the YCbCr color space and down samples the chroma channels. Images are then split into 8 × 8 blocks and each block is transformed with a two-dimensional discrete cosine transform (DCT),followed by a quantization of DCT coefficients. More details of JPEG compression algorithms can be found in [32]. Unpleasing block artifacts are usually introduced by the JPEG compression.The quality of compressed images is determined by a quality factor q ∈ [0, 100], where a lower q indicates a higher compression ratio and worse quality. We use the Py-Torch implementation - DiffJPEG [33].

Faster R-CNN

Following the development of R-CNN and Fast R-CNN, Ross B. Girshick proposed Fast R-CNN in 2016. Overall performance is far better, particularly when it comes to detecting speed. Faster R-CNN reduces the number of proposed frames from roughly 2000 to about 300 by ingeniously using the convolution network to build the proposed box and sharing it with the object detection network. Additionally, the suggested box is of higher quality. When compared to Fast RCNN, the CNN of object detection shares characteristics with the CNN of suggestion window, and RPN (Region Proposal Network) generates the recommendation window instead of the original Selective Search technique.[10]

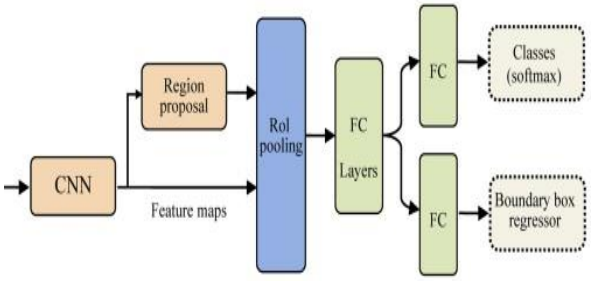


Fig 4. Faster R-CNN architecture [7]

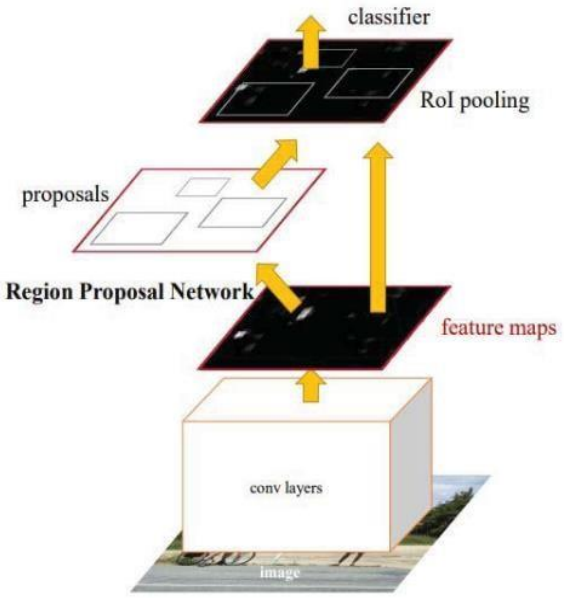


Fig 5.Faster R-CNN [6]

Convolution Layers

The input image is passed through a Convolutional Neural Network (CNN). This CNN extracts features from the image while preserving its spatial information. The output of this process is a feature map, which essentially represents the image at different scales and levels of abstraction. Faster R-CNN architecture consists of two modules [7]

1. Region Proposal Network (RPN)
2. Fast R-CNN detector [7,8]

1.**Region Proposal Network (RPN)** RPN is composed of neural networks such as convolutional layer and fully-connected layer, so learning is possible. In addition, fast calculation is possible by using GPU operation. RPN receives a 256-dimensional or 512-dimensional feature vector from the feature extractor and creates an intermediate layer through the sliding window. And then, it convolutions into a classifier layer and a regressor layer. The classifier layer applies 1×1 filter to get the output. In this layer, k anchor box with various scales and ratios is generated through the sliding window, and two scores indicating the existence of objects are assigned. 1×1 filter is also applied to the regressor layer, k anchor boxes are generated, and 4 coordinate values for displaying the coordinates of the bounding box are assigned.[7]

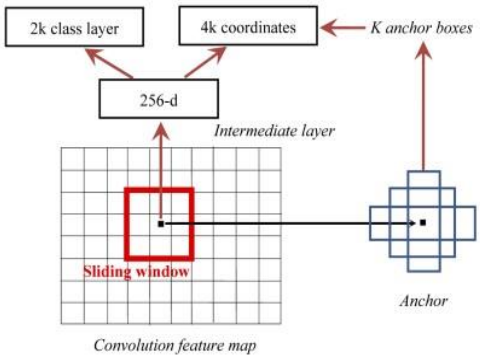


Fig 6. Architecture of Region Proposal Network [7]

Anchor boxes

We simultaneously forecast numerous region proposals at each sliding-window location, where k is the maximum number of potential proposals for each site. Accordingly, the cls layer produces 2k scores that estimate the probability of an item or not for each proposal, whereas the regression layer produces 4k outputs storing the locations of k boxes [4]. The k proposals are parameterized with respect to k anchors, or reference boxes. An anchor with a scale and aspect ratio is centered at the problematic sliding window (Figure 6). Three scales and three aspect ratios are used by default, resulting in k = 9 anchors at each sliding position. There are W Hk for a convolutional feature map with a size of W × H, which is around 2,400.

Fig 7. Example detections using RPN proposals on PASCAL VOC 2007 test. [6]

2.Fast R-CNN detector

A series of item proposals and the complete image are fed into a Fast R-CNN network. To create a convolution feature map, the network first processes the entire image through many max pooling and convolutional (conv) layers. Next, from the feature map, a fixed-length feature vector is extracted via a region of interest (RoI) pooling layer for every object proposition.Every feature vector feeds into a series of fully connected (fc) layers, which eventually split into two sibling output layers: one that generates four real-valued numbers for each of the K object classes, and another that generates softmax probability estimates over K object classes plus a catch-all "background" class. For one of the K classes, each set of four values encodes refined bounding-box coordinates [8].

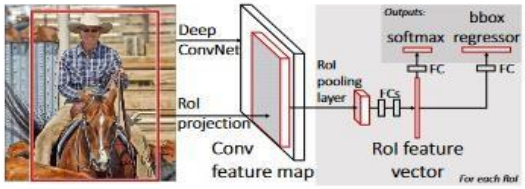


Fig 8. Fast R-CNN architecture

RoI (Region of Interest) Pooling layer

The features inside any eligible region of interest are converted by the RoI pooling layer using max pooling into a tiny feature map with a set spatial extent of H × W (e.g., 7 × 7), where H and W are layer hyper-parameters that are independent of any specific RoI. The ROI in this paper is a rectangular window into a convolutional feature map. A four-tuple (r, c, h, w) that describes each RoI's height (h, w), breadth (h, c), and top-left corner (r, c) defines each RoI. The process of ROI max pooling involves splitting the h × w RoI window into a H × W grid of subwindows with approximate sizes of h/H × w/W. Each subwindow's values are then max-pooled into the corresponding output grid cell. Each feature map channel is subjected to pooling independently, as in the conventional system given in [16].

Object Classification

Classification layers use the proposal feature maps to calculate the proposal's class, and bounding box regression toget the final exact position of the checkbox [15]. Classification layers get the 7×7 = 49 size of the proposal feature maps fromthe Roi Pooling layers, and calculate which category each proposal and specifically judge of belonging to[16] (such as people, cars, horses, etc.) through the full connected layer with softmax and a output class probability vector can be obtained.Classification layers use the bounding box regression again to get the position offset bbox_pred for each proposal, and returns more accurate target detection box. To obtain a more accurate rect box, part of the network structure of classification layers is shown in Fig 9. [10]

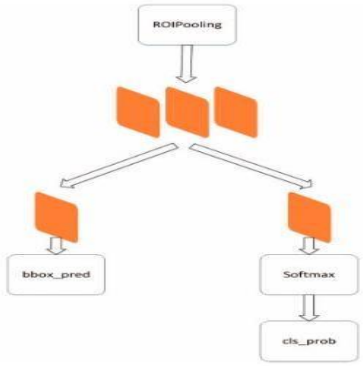


Fig 9: A portion of the classification layer's network structure [10]

Bounding Box Regression:

Bounding box regression is performed to refine the proposed bounding boxes. The model predicts offsets that adjust the dimensions and position of the bounding boxes, improving their alignment

with the objects present in the image. For bounding box regression, we adopt the parameterizations of the 4 coordinates following [6]:

$$
\begin{aligned}
t_{\mathrm{x}} &= (x - x_{\mathrm{a}})/w_{\mathrm{a}}, \quad t_{\mathrm{y}} = (y - y_{\mathrm{a}})/h_{\mathrm{a}}, \\
t_{\mathrm{w}} &= \log(w/w_{\mathrm{a}}), \quad t_{\mathrm{h}} = \log(h/h_{\mathrm{a}}), \\
t^*_{\mathrm{x}} &= (x^* - x_{\mathrm{a}})/w_{\mathrm{a}}, \quad t^*_{\mathrm{y}} = (y^* - y_{\mathrm{a}})/h_{\mathrm{a}}, \\
t^*_{\mathrm{w}} &= \log(w^*/w_{\mathrm{a}}), \quad t^*_{\mathrm{h}} = \log(h^*/h_{\mathrm{a}}),
\end{aligned}
\qquad [6]
$$

where the box's width, height, and center coordinates are indicated by the variables x, y, w, and h. The ground truth box, anchor box, and anticipated box are represented by the variables x, xa, and x*, respectively (likewise for y, w, h). Bounding-box regression from an anchor box to a neighboring ground-truth box can be applied to this.

4. IMPLEMENTATION

The general framework of this research is shown in Fig.10.

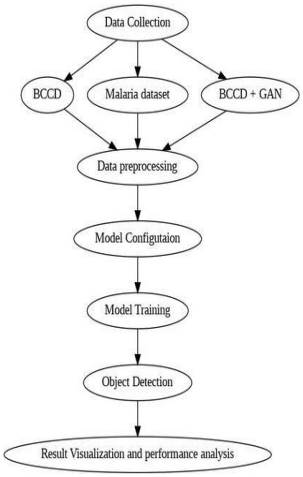


Fig 10.General Framework

1) Data collection

The first part of the whole procedure is the data collection. Data collection is the process of gathering information from various sources for a specific research objective which serves as the foundation for training and evaluating the machine learning model.

For this step we gather 2 datasets namely BCCD and Malaria.

a) BCCD Dataset

A small-scale dataset for blood cell detection is the Blood Cell Count and Detection (BCCD) dataset is taken from [18] This dataset contains the images of the blood samples (blood cell smear images) taken from the microscopic slides including RBCs (Erythrocytes),WBCs(Leukocytes) and Platelets (Thrombocytes).The dataset may include images using microscopy staining techniques like Giemsa or Wright's, which enhance the visibility of blood cell structures. Images are typically annotated with bounding boxes or segmentation masks to indicate the locations and boundaries of individual blood cells within the image. For our

project implementation the BCCD dataset includes 364 images (in jpg format) along with annotations in xml format.

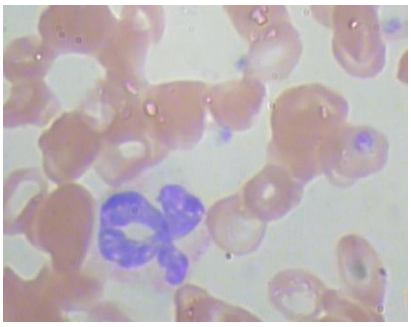


Fig 11. Sample image from BCCD Dataset

A sample image from the BCCD dataset is displayed in Figure 11, and a sample xml annotation file related to the BCCD dataset is highlighted in Figure 9.

As seen in figure 12, the 640x480 pixel RGB image has three channels, three labeled objects—two red blood cell (RBC) examples and one white blood cell (WBC) sample. The positions and measurements of the items are indicated by bounding boxes. Bounding boxes identify the regions of interest for each type of cell in the image; the WBCs are located at coordinates (127, 40), the RBCs at coordinates (317, 93), and the WBCs at coordinates (379, 146).

```
▼<annotation>
   <folder>JPEGImages</folder>
   <filename>BloodImage_00003.jpg</filename>
   <path>/home/pi/detection_dataset/JPEGImages/BloodImage_00003.jpg</path>
  ▼<source>
     <database>Unknown</database>
   </source>
  ▼<size>
     <width>640</width>
     <height>480</height>
     <depth>3</depth>
   </size>
   <segmented>0</segmented>
  ▼<object>
     <name>WBC</name>
     <pose>Unspecified</pose>
     <truncated>0</truncated>
     <difficult>0</difficult>
    ▼<bndbox>
       <xmin>127</xmin>
       <ymin>40</ymin>
       <xmax>344</xmax>
       <ymax>226</ymax>
     </bndbox>
   </object>
  ▼<object>
     <name>RBC</name>
     <pose>Unspecified</pose>
     <truncated>0</truncated>
     <difficult>0</difficult>
    ▼<bndbox>
       <xmin>317</xmin>
       <ymin>93</ymin>
       <xmax>424</xmax>
       <ymax>195</ymax>
     </bndbox>
   </object>
  ▼<object>
     <name>RBC</name>
     <pose>Unspecified</pose>
     <truncated>0</truncated>
     <difficult>0</difficult>
    ▼<bndbox>
       <xmin>379</xmin>
       <ymin>146</ymin>
       <xmax>471</xmax>
       <ymax>234</ymax>
     </bndbox>
   </object>
```

Fig 12. BCCD dataset sample annotation xml file

b) Malaria Dataset

The malaria dataset is a collection of images captured from blood smears or thin films, showing blood cells infected with Plasmodium vivax, a common type of malaria. It includes various stages of the parasite's lifecycle and can be annotated with bounding boxes or segmentation masks. We can use this dataset to develop and evaluate machine learning models for automated malaria detection, aiding in early diagnosis and treatment.

Similarly, the Malaria dataset includes 1182 images (in jpg format) along with annotations in xml format. Fig 13 shows a sample image from malaria dataset and Fig 11 highlights a sample annotation file in xml format associated with the Malaria dataset. [19]

Fig 13. Sample image from Malaria Dataset

The annotation file, as seen in fig. 14, shows the locations of four Plasmodium parasite occurrences in an image that was obtained from the Makerere Automated Lab Diagnostics Database and microscopically captured from Mulago National Referral

Hospital.

The location of each bounding box is defined using four coordinates—x_{min}, y_{min}, x_{max}, and y_{max}. The top, left, right, and bottom edges of each box are delineated by these limits. The following are the four Plasmodium cases:-

Instance 1: Type of label: Plasmodium Location: (617.25, 175.73, 657.25, 215.73). Instance2:

Type of label: Plasmodium
Location: (783.40, 66.78, 743.40, 106.78).

Instance2: Type of label: Plasmodium
Location: (476.62, 488.66, 516.62, 528.66)

Instance 4: Plasmodium label

By using these coordinates, one can pinpoint the Plasmodium instances in the image and examine the parasite's distribution and amount.

c) BCCD + GAN Dataset

The BCCD dataset undergoes two types of GANs to generate good quality images. First type is the GAN to generate new images and the second type is the Real-ESRGAN to enhance the quality of the generated images.

i) **GAN to generate new images**

By using the BCCD dataset in a GAN architecture, fresh artificial images of platelets and blood cells can be created. The GAN gains the ability to capture the minute details and variances found in blood cell morphology, such as size, shape, and staining

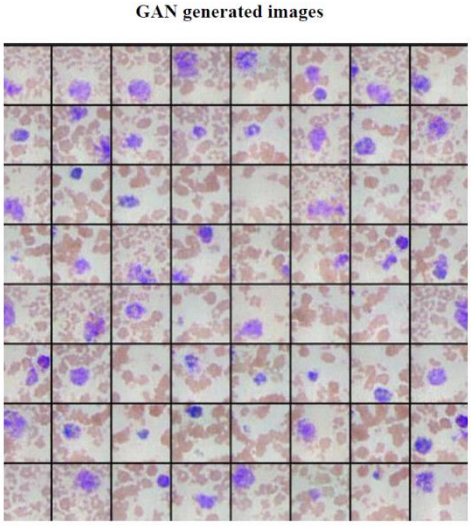


Fig 15. GAN generated images

Figure 15 showcases the 64 images generated by the Generative Adversarial Network (GAN) after undergoing 1500 epochs of training with a learning rate of 0.002 [6]. The input images utilized to train the GAN system originate from the Blood Cell Count and Detection (BCCD) dataset, which is sourced from reference [18].

The GAN's images demonstrate the model's development as well as its ability to create lifelike depictions of platelets and blood cells. One can gain insights on the GAN's capacity to capture the richness and diversity of blood cell shape by tracking the development of these generated images during training.

ii)Real-ESRGAN for enhancement of images The GAN generated images undergo Real-ESRGAN to serve the purpose of increasing the resolution . One of the most important aspect of object detecting is the requirement of high quality images. Providing mediocre quality images to the Object detection model will lead to mismatches and hinderance in the detection mechanism. Real-ESRGAN comes in handy when it comes to tasks like improving the image quality and produce HR (High Resolution) images.

The project deals with encouraging the usage of higher quality images with the help of Real-ESRGAN. The images generated using GANs lack clarity and are low in resolution. The RealESRGAN comes in handy for resolving issues with quality of the images

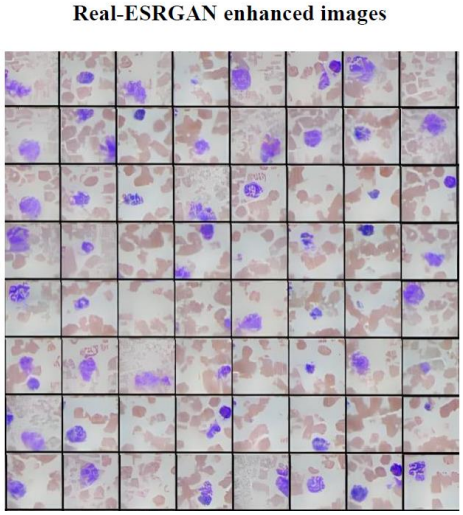


Fig 16. Real-ESRGAN enhanced images

Fig 16 presented illustrates the set of 64 images generated by the Generative Adversarial Network (GAN) undergoing the Real-ESRGAN procedure, aimed at enhancing their quality. Real-ESRGAN is applied to elevate the resolution of these images by a factor of 4X, significantly improving their visual fidelity and detail.

Every one of the first-generated images is transformed by the RealESRGAN procedure to improve resolution while maintaining important details and features. By upscaling the photographs, any possible problems with pixelation or blurriness that may have existed in the original generated images are efficiently addressed and a better level of clarity and sharpness is achieved in the final product.

By subjecting the images to a 4X resolution upgrade via RealESRGAN, the resulting outputs exhibit a noticeable improvement in quality, making them better suited for tasks requiring fine details and high visual fidelity.

2) Data Pre-processing

The goal of data pre-processing is to clean, standardize, and prepare data for consistent, accurate, and compatible analytical methods. It involves transforming raw data into a format appropriate for analysis, machine learning, or other data science tasks. Pre-processing of the BCCD and Malaria datasets is something we can perform.

Pre-processing the BCCD dataset comprises cleaning, transcoding, and getting ready images for analysis or training machine learning models. This improves model performance. This guarantees precise detection and classification of plasmodium and blood cells.

The BCCD and malaria datasets require effective model processing techniques, such as image scaling, brightness adjustments, noise reduction, normalization, color space conversion, data augmentation, and annotation management, for improved performance and training.

3) Model Configuration (Faster R-CNN)

After pre-processing the data, Faster R-CNN can be used to carry out object detection tasks in the BCCD and Malaria datasets.

The Region Proposal Network (RPN), Detection Network, and Feature Network are the three neural networks that make up Faster-RCNN. The input image's shape and structure are preserved as feature maps are created by the feature network. Bounding box regression, classification, and region of interest (RoI) generation are the functions of RPN's three convolutional layers. RPN generates about 2,000 region ideas, of which the top N are utilized for testing.

The final class and bounding boxes are produced by the Detection Network, which consists of four fully linked layers. Features are cropped in order to categorize the bounding box inthe bounding box regression and classification layers, which share two common layers. Setting up and fine-tuning the network architecture, hyper parameters, and dataset-specific input/output formats are all part of the model configuration process.

For all the three datasets, the configuration operates as follows:

Network Structure:

The network architecture consists of a base CNN for feature extraction, a Region Proposal Network (RPN) [6] for blood cell source recommendations, and ROI Pooling for regression and classification. ROI Pooling pulls uniform-sized features from each region for these purposes.

Hyper parameters:

Learning rate, batch size, IoU thresholds, and loss functions for model training, regression, and classification are examples of hyper parameters. These parameters control the speed at which the model is trained, ascertain the stability of the model and its resources, locate positive samples, and apply the proper loss functions.

Learning Rate:

During training, the learning rate dictates how much the model's weights are updated. The model may converge too quickly and miss the ideal solution if the learning rate is too high, whereas it may converge slowly if the learning rate is too low. When training with the BCCD dataset, a learning rate between 0.0001 and 0.01 is typical.

Batch Size:

The quantity of samples handled concurrently during training is referred to as the batch size. Although a bigger batch size may demand more memory, it can speed up training. The number of samples the model processes in each iteration from the BCCD dataset depends on the batch-size. Depending on hardware capabilities, batch sizes ranging from 16 to 64 are commonly utilized.

IoU Thresholds:

A projected bounding box's overlap with a ground truth bounding box is measured using an intersection over union, or IoU.

A predicted bounding box's status as a true positive is determined by its intersection with union (IoU) thresholds. A typical IoUthreshold for Faster R-CNN on the BCCD dataset is approximately 0.5.

$$\text{IoU} = \frac{\text{Area of Intersection}}{}$$

Area of Union **Loss Functions:**

During training, the difference between expected and actual values is measured in Faster R-CNN using loss functions. Regression loss and classification loss are combined to get the total loss function:

Total Loss= Classification Loss + Regression Loss

You can combine classification loss (such cross-entropy loss) with object recognition tasks in the BCCD dataset. These losses aid in directing the model's ability to produce precise forecasts for various blood cell types.

Input and Output:

Pre-processed images are matched with the input size, blood cell kinds are specified by the output classes, and the model output format is defined by the output format.

Training and Validation:

While the underlying CNN employs pre-trained weights, techniques like as data splitting, data augmentation, and transfer learning are utilized to divide datasets for testing, training, and validation.

4) Model Training

Using pre-processed training data, the configured Faster RCNN[6] is used to train the model to identify and categorize malaria parasites and components of blood cells.

Pre-processed images and their ground truth annotations are loaded, and the images are then sent through an RPN and a basic CNN to produce region suggestions and process features. Ground reality is compared with model forecasts to calculate losses namely the generator loss and the discriminator loss.

By combining computed losses, backpropagation is a technique that determines the gradients and the loss depending on model parameters such as weights and biases.

Through processing several batches of data across several epochs, learning from each epoch, and updating its parameters, the model goes through an iterative training process. Optimization methods are used for parameter changes. The model's performance is evaluated through validation utilizing metrics such as mean Average Precision.

Additionally, it employs early stopping to avoid overfitting and stores checkpoints during training to maintain progress. Training is also stopped when validation performance no longer improves. During an epoch, the model goes through each training batch of data, makes any necessary adjustments to its parameters, and then restarts from the beginning of the dataset. Training and validation accuracy usually rise with the number of epochs, and some initial gains are noticeable. Fig 17 represents the start of the training process and Fig 18 represents the end of the training process.

We can also see the average batch loss (the average loss calculated over a batch of data during training) and Total loss (combined loss calculated across all batches in an epoch during training) obtained during the training process.

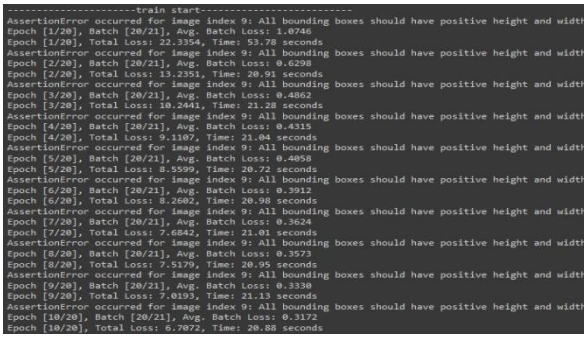


Fig 17. Start of training process

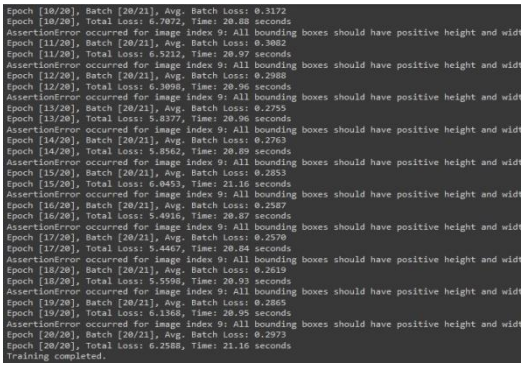


Fig 18. End of the training process

5) Object detection

One type of computer vision problem is object detection, which is locating and identifying particular things in an image. The trained model can identify several blood cell types (RBCs, WBCs, and platelets) as well as malaria parasites when working with blood cell images, such as those found in the BCCD and Malaria datasets.

Faster R-CNN Object Detection:

- Model Overview

By merging convolutional neural networks (CNN) and region proposal networks (RPN), the state-of-the-art object identification model known as Faster-CNN (Region-based Convolutional Neural Network) accurately recognizes and classes objects in images. The model is trained using blood smear images from the Malaria and BCCD datasets, and it can also be enhanced by synthetic images generated by GANs.

- Image Input:

Fresh blood smear pictures are fed into the Faster R-CNN [34] model that has been trained for analysis. These pictures may include malarial parasites (P. Vivax) in addition to different blood cell components (RBCs, WBCs, and platelets).

- Region Proposal Network (RPN)[6]:

Region proposals are portions of the image[6] that are probably going to have interesting things in them. By helping the model concentrate its attention on regions that may contain objects, these region suggestions enhance the effectiveness and precision of the detection procedure.

- Feature Extraction:

Features are extracted from the image by the CNN component of the Faster R-CNN model. Important details regarding the contents of various image regions are provided by these features.

- Region of Interest (RoI) [6] Pooling:

An ROI pooling layer [6] is applied to the region suggestions produced by the RPN. This layer enables the model to accommodate varying area sizes by extracting fixed-size feature maps for every region proposal.

- Classification and Bounding Boxes:

The model classifies each region as belonging to one of the following classes: RBC, WBC, platelets, malarial parasite, or none at all, based on the RoI-pooled properties. For every object found inside the regions of interest, the model further forecasts the bounding boxes, which include size and location.

- Output:

For every object in the image that is detected, the model outputs the bounding boxes and its classification. The categories show if something is a malarial parasite, platelet, WBC, RBC, or none of these. The bounding boxes give the size and coordinates of any items that have been identified in the picture.

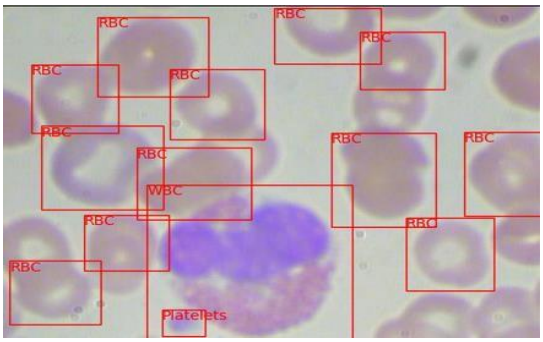


Fig 19. Classification of Blood cell images

6) Result visualization and Performance analysis

The result visualization and performance analysis deals with the generation of the output images along with their bounding boxes and labels. The results in the detection of RBC, WBC and platelets consists of bounding boxes around the entities and the labels being allocated to them. Results in this section also mentions the number of RBCs, WBCs and platelets in the provided dataset.The performance analysis, here too revolves accuracy parameters like mAP and AP. Visualization of obtained results is through graphical analysis.

While in the detection of the malarial parasite (Plasmodium vivax), the results here also consist of bounding boxes around the plasmodium and labels.It's critical to assess the model's performance after training and visualize the outcomes.

To evaluate the accuracy and dependability of the model, a number of evaluation indicators can be employed. These indicators direct future development by illuminating the model's advantages and disadvantages. The main evaluation indicators are as follows:

Evaluation Indicators

Accuracy: It indicates the proportion of the correctly detected samples to the total number of [34] samples detected. The recall rate indicates the proportion of the correctly detected samples in all samples that should be detected.

a) Accuracy is defined as:

$$Accuracy = \frac{TP + TN}{TP + FN + TN + FP} \quad [17]$$

b) Recall: Recall calculates the percentage of real positive detections in the dataset that are true positive detections. A high recall value means that the majority of real events are effectively detected by the model.

$$Recall = \frac{TP}{TP + FN} \quad [17]$$

Where:

a) True Positive (TP) is the positive prediction of positive data(e.g., detected a P. vivax parasite or blood cell type).
b)True Negative (TN):A true negative value is an actual negative estimate of data(e.g., correctly identified a region as not containing a blood cell type or P. vivax parasite).
c) False Positive (FP):A false positive value is the incorrect prediction of positive data(e.g., falsely detected a blood cell type or P. vivax parasite).
d) False Negative(FN):False negative value is a negative estimate of actual positive data(e.g., missed detecting a blood cell type or P. vivax parasite).

It is being discussed in detail in the below ‘Results’ section.

c) Average Precision(AP)

It assesses how well the model balances precision and recall for that class at different confidence levels. The area under the precision-recall curve for a given class is used to compute AP. The precision-recall curve is frequently numerically integrated to approximate it.

d) Mean Average Precision(mAP) [34] mAP gives an overall assessment of the model's performance in an item detection job across all classes. It provides a thorough understanding of the model's object detection accuracy by representing the average of the AP values for every class.

5. RESULTS

The outcomes are displayed via graphic figures that make use of an object detection technology. In the BCCD and BCCD + GAN datasets, the system recognizes important blood components such as red blood cells (RBCs), white blood cells (WBCs), platelets, and the malaria-causing Plasmodium Vivax (P Vivax). The identified zones are displayed in each figure with the corresponding classification groups.

a)BCCD cells detection results In Fig. 15, a test image displays RBCs, WBCs and Platelets detected by the system. The figure compares two sets of data, labeled as "Target" and "Prediction," which represent different measurements of blood components, specifically RBC (coded as 1), WBC (coded as 2) and platelets (coded as 3).

For instance, the "Target" data in the picture displays a series of 1s and 2s, where 1 denotes the red blood cell and 2 the white blood cell. A 2 in the second place, a 3 in the thirteenth position, and 1 in every other position in the "Prediction" statistics indicate a greater RBC count than WBCs and platelets.

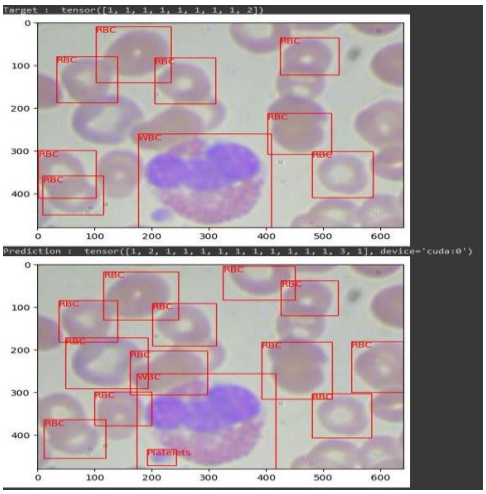


Fig 20. Component classification detection

Fig 20 highlights the count of the blood cell components (RBCs, WBC's and Platelets)

```
rbc_label = 1  # Assuming RBC label is 0
wbc_label = 2  # Assuming WBC label is 1
platelets_label = 3  # Assuming platelets label is 2

# Initialize counters
rbc_count = 0
wbc_count = 0
platelets_count = 0

# Iterate over predicted labels
for label in pred_labels:
    if label == rbc_label:
        rbc_count += 1
    elif label == wbc_label:
        wbc_count += 1
    elif label == platelets_label:
        platelets_count += 1

print("Count of RBC:", rbc_count)
print("Count of WBC:", wbc_count)
print("Count of Platelets:", platelets_count)
```

```
Count of RBC: 265
Count of WBC: 22
Count of Platelets: 14
```

Fig 21. Count of the blood cell components

Fig. 21 shows the mAP (Mean Average Precision) and AP(Average Precision) values for the blood cell components. The mean average precision, or mAP for short, is a single number that represents the average performance of several classes' worth of AP values. It is used to evaluate an object recognition model's overall effectiveness.

Average Precision, or AP for short, is a measure of how well an object identification model performs overall for a given class by calculating the area under the precision-recall curve. The obtained mAP is 92% for RBC, WBC and Platelets altogether. The AP(Average Precision) is 88.45%, 99.35% and 88.59% for RBC,WBC and Platelets as shown in Fig 22.

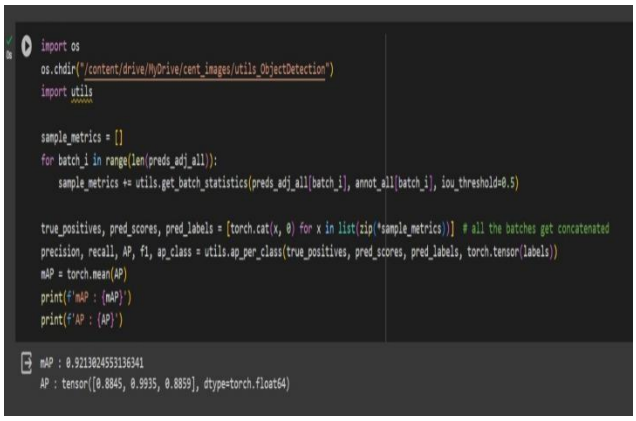


Fig 22. mAP and AP values for Blood cell components

Fig. 23 shows the AP comparison between Training and Dataset for Blood cell components (RBC,WBC and Platelets).The solid lines represent the training and the dotted lines represent the dataset for the blood cell components.

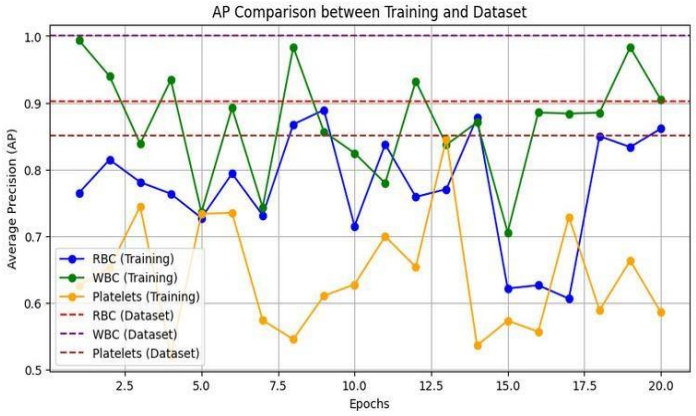


Fig 23. AP Comparison between Training and Dataset for blood cell Components

Fig 24 shows the Scatter plot of the blood cell components (RBC, WBC and Platelets).

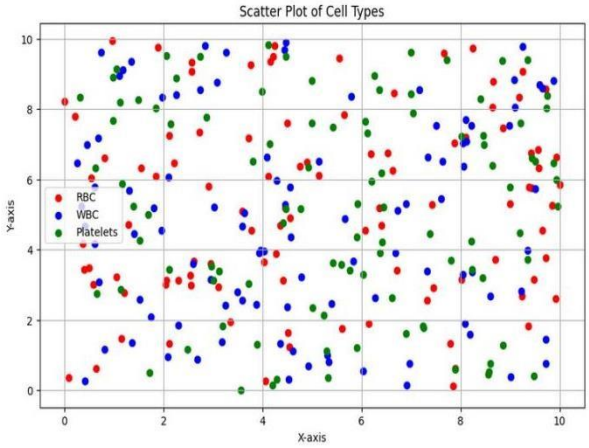


Fig 24. Scatter Plot of Cell types

(b) Malarial parasite (Plasmodium vivax) detection results In Fig. 20, a test image displays Plasmodium Vivax (P Vivax) species detected by the system. The figure compares two sets of data, labeled as "Target" and "Prediction.

In the figure, the "Target" data and the "Prediction" data shows the sequence of 1s thereby indicating the presence of the Malaria causing Plasmodium Vivax species.

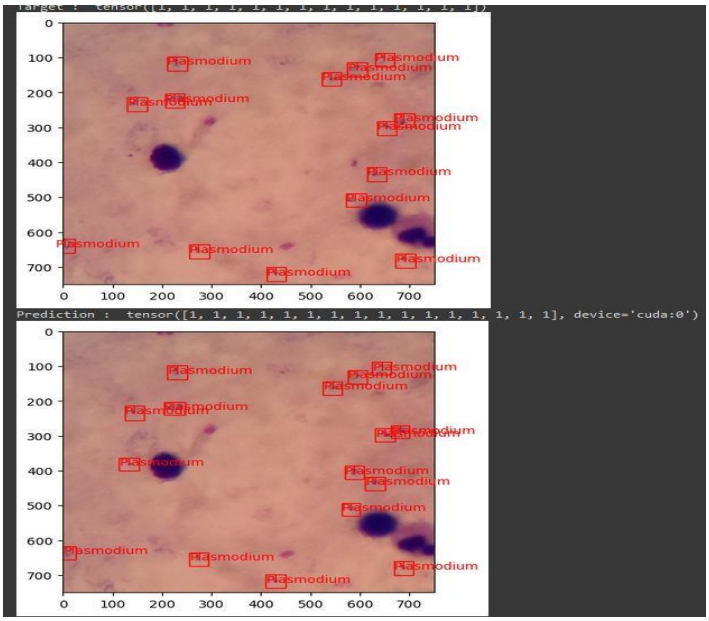


Fig 25. Detection of Plasmodium Vivax

Fig. 26 shows the mAP (Mean Average Precision) and AP (Average Precision) values for the detection of P.Vivax. The obtained mAP and AP (Average Precision) is 73%.

```
true_positives, pred_scores, pred_labels = [torch.cat(x, 0) for x in list(zip(*sample_metrics))]  # all the batches get concatenated
precision, recall, AP, f1, ap_class = utils.ap_per_class(true_positives, pred_scores, pred_labels, torch.tensor(labels))
mAP = torch.mean(AP)
print(f'mAP : {mAP}')
print(f'AP : {AP}')
```

```
mAP : 0.7300271757000063
AP : tensor([0.7300], dtype=torch.float64)
```

Fig 26. mAP and AP values for P.Vivax

Fig. 27 shows the AP comparison between Training and Dataset for Malaria Detection. The solid line represent the training and the dotted line represent the dataset for detection of P.Vivax species.

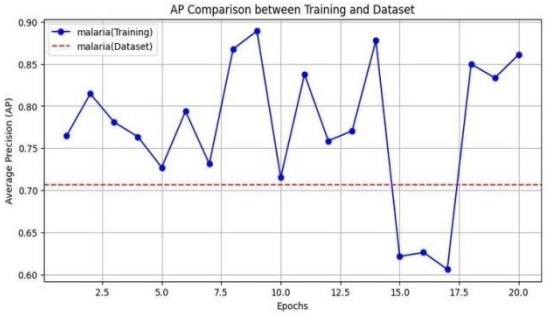


Fig 27. AP Comparison between Training and Dataset for Malaria Detection

(c) BCCD + GAN enhanced detection results

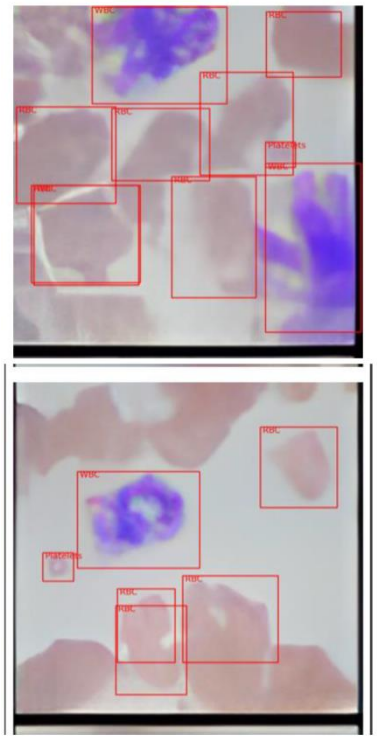


Fig 28. Detection of GAN enhanced blood cells

The test image in Fig 28 shows the entities RBCs, WBCs and platelets that the Faster R-CNN model has identified.

6. CONCLUSION AND FUTURE RECOMMENDATIONS

In conclusion, our project successfully implemented the Faster R-CNN architecture across three distinct datasets: the Blood Cell Count and Detection (BCCD) dataset, a malarial dataset, and a composite dataset that includes original BCCD data supplemented with GAN-generated images. This multifaceted approach has significantly enhanced the model's ability to precisely detect and classify various blood cell types and Plasmodium vivax parasites. By utilizing GANs to augment the BCCD dataset, we addressed the challenge of limited data, thereby improving the detection accuracy and reliability of the system, crucial for supporting real-world medical diagnostics.

Looking forward, enhancing this project could involve expanding the dataset with images from diverse populations to increase model robustness across different patient groups.

Additionally, integrating advanced machine learning strategies like semi-supervised learning could capitalize on unlabeled data, refining model accuracy. Developing real-time analysis capabilities and exploring newer neural network architectures or advanced GANs would further optimize performance and training efficiency. Such advancements promise significant contributions to medical diagnostics, particularly in the early detection and management of malaria and related blood disorders.

REFERENCES

[1] IhtishamUlHaq, Umar Sadique, Shahzad Anwar and Muhammad Tahir Khan"An Intelligent Approach for Blood Cell Detection Employing Faster RCNN",2023

[2] Rose Nakasi,Ernest Mwebaze,Aminah Zawedde,Jerem y Tusubira,· Benjamin Akera,Gilbert Maiga"A new approach for microscopic diagnosis of malaria parasites in thick blood smears using pre-trained deep learning models"

[3] Rogelio RuzckoTobias,Luigi Carlo De Jesus, Matt Ervin Mital, Marielet Guillermo "Faster RCNN Model With Momentum Optimizer for RBC and WBC Variants Classification"

[4] Sneha Raina, AbhaKhandelwal, Saloni Gupta, AlkaLeekha "Blood Cells Detection Using Faster- RCNN"

[5] Miss. Priyanka L. Khambayat, Dr. Dinesh D. Patil, Prof. Yogesh S. Patil "IMAGE PROCESSING TECHNIQUES TO IDENTIFY RED BLOOD CELLS"

[6] Shaoqing Ren, Kaiming He, Ross Girshick, Jian Sun "Faster R-CNN: Towards Real Time Object Detection with Region Proposal Networks", 2016

[7] Jongchol Kim, Jiyong Kim and JinmyongRi "Generative Adversarial networks and faster-region convolutional neural networks based object detection in X-ray baggage security image".

[8] Ross Girshick, "Fast R-CNN"

[9] M.Sushma Sri, B. RajendraNaik, K. Jaya Sankar "Object Detection based on Faster R-CNN", 2021

[10] Bin Liu, Wencang Zao and Qiaoqiao "Study of Object Detection Based on Faster R-CNN"

[11] Jane Hung, Anne Carpenter "Applying Faster R-CNN for Object Detection on Malaria Images"

[12] A.I. Shahin, YanhuiGuo , K.M. Amin, Amr A. Sharawi"White blood cells identification system based on convolutional deep neural learning networks"

[13] Jane Hung,Allen Goodman, Stefanie Lopes,Gabriel Rangel, Deepali Ravel, Fabio Costa "Applying Faster R-CNN for Object Detection on Malaria Images"

[14] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. "Imagenet classification with deep convolutional neural networks. In Advances in neural information processing systems"

[15] He K, Zhang X, Ren S, et al, "Deep Residual Learning for ImageRecognition," 2015

[16] He, X. Zhang, S. Ren, and J. Sun. "Spatial pyramid poolingin deep convolutional networks for visual recognition", 2014

[17] An Improved Faster R-CNN for Small Object Detect19ion CHANGQING CAO, BO WANG , WENRUI ZHANG, XIAODONG ZENG, XU YAN, ZHEJUN FENG, YUTAO LIU , AND ZENGYAN WU-2019
[18] Shengan BCCD dataset
[19] AnandKoirala, MeenaJha, Srinivas Bodapati, Animesh Mishra, GirijaChetty, Praveen Kishore Sahu, SanjibMohanty, TimirKantaPadhan, JyothiMattoo and AjatHukkoo, "Deep learning for Real-Time Malaria Parasite Detection and counting using YOLO-mp"
[20] Zhengwei Wang, Qi She, Tom ´as E. Ward, "Generative Adversarial Networks in Computer Vision: A Survey and Taxonomy"
[21] Liang Gonog and Yimin Zhou, "A Review: Generative Adversarial Networks"
[22] Ian J. Goodfellow, Jean Pouget-Abadie∗, Mehdi Mirza, Bing Xu, David Warde-Farley, SherjilOzair† , Aaron Courville, YoshuaBengio, "Generative Adversarial Nets"
[23] Alankrita Aggarwal , Mamta Mittal , GopiBattineni, "Generative adversarial network: An overview of theory and applications"
[24] Li Ma1 · Renjun Shuai1 · Xuming Ran2 · Wenjia Liu3 · Chao Ye1, "Combining DC-GAN with ResNet for blood cell image classification"
[25] Gadelha M, Maji S, Wang R (2017) 3D shape induction from 2D views of multiple objects. In: 2017 International Conference on 3D Vision (3DV). IEEE, pp 402–411
[26] Mathieu M, Couprie C, LeCun Y (2015) Deep multi-scale video prediction beyond mean square error. arXiv:1511.05440
[27] Reed S, Akata Z, Yan X, Logeswaran L, Schiele B, Lee H (2016) Generative adversarial text to image synthesis. arXiv:1605.05396
[28] Iizuka S, Simo-Serra E, Ishikawa H (2017) Globally and locally consistent image completion. ACM Transactions on Graphics (ToG) 36(4):107
[29] SukruBurak Cetin, "Real-ESRGAN: A deep learning approach for general image restoration and its application to aerial images"
[30] ZHENGWEI ZHU, YUSHI LEI1, YILIN QIN, CHENYANG ZHU, AND YANPING ZHU, "IRE: Improved Image Super-Resolution Based on RealESRGAN"
[31] Xintao Wang, LiangbinXie, Chao Dong, Ying Shan, "Real-ESRGAN: Training Real-World Blind Super-Resolution with Pure Synthetic Data"
[32] Richard Shin and Dawn Song. "Jpeg-resistant adversarial images." In NeurIPS Workshop on Machine Learning and Computer Security, 2017
[33] Michael R Lomnitz. Diffjpeg. https://github.com/mlomnitz/DiffJPEG, 2021.
[34] Bojia Qiu, "Food Recognition and Nutrition Analysis using Deep CNNs", 2019