Shahid Beheshti University

Artificial Neural Networks

M.Sc - Fall 2024

Assignment 2 - Part 2

Lacuna Malaria Detection

Author:
Zahra Mohammad Beigi

Student Number:
402422144

November 8, 2024

Contents

1 Task Overview

The objective of this task is to build a Convolutional Neural Network (CNN) to classify microscopic images of blood samples into three classes:

- NEG (Negative for Malaria)
- Trophozoite (Malaria parasite stage)
- WBC (White Blood Cells)

In addition to classification, we are going to predict the bounding box coordinates (`xmin, xmax, ymin, ymax`) of the objects within the images, making this a combined classification and regression task.

2 Dataset Overview

The dataset consists of a total of 23,530 entries, each representing a microscopic image of a blood sample. The data includes information necessary for both classification and bounding box regression tasks. Table 1 provides an overview of the dataset structure.

Column	Data Type	Description
`Image_ID`	Object	Unique identifier for each image
`class`	Object	Label of the sample, categorized as *NEG*, *Trophozoite*, or *WBC*
`confidence`	Float64	Confidence score associated with the class label
`ymin`	Int64	Minimum y-coordinate of the bounding box
`xmin`	Int64	Minimum x-coordinate of the bounding box
`ymax`	Int64	Maximum y-coordinate of the bounding box
`xmax`	Int64	Maximum x-coordinate of the bounding box
`path`	Object	File path to the image

Table 1: Overview of the Dataset

The dataset occupies approximately 1.4 MB in memory, with data types comprising object, float64, and int64 types. The bounding box coordinates (`ymin`, `xmin`, `ymax`, `xmax`) provide the necessary information for localizing objects within each image, facilitating the regression task alongside classification.

3 Exploratory Data Analysis (EDA)

3.1 Bounding Box and Class Distribution

This section focuses on understanding the distribution of bounding boxes per image, the class distribution of these bounding boxes, and their occurrences across unique images.

Bounding Box Counts per Image

Figure 1 summarizes the distribution of bounding boxes per image. The dataset contains 2,747 unique images, each with an average of 8.57 bounding boxes. The number of bounding boxes per image varies significantly, with a minimum of 1 and a maximum of 92.

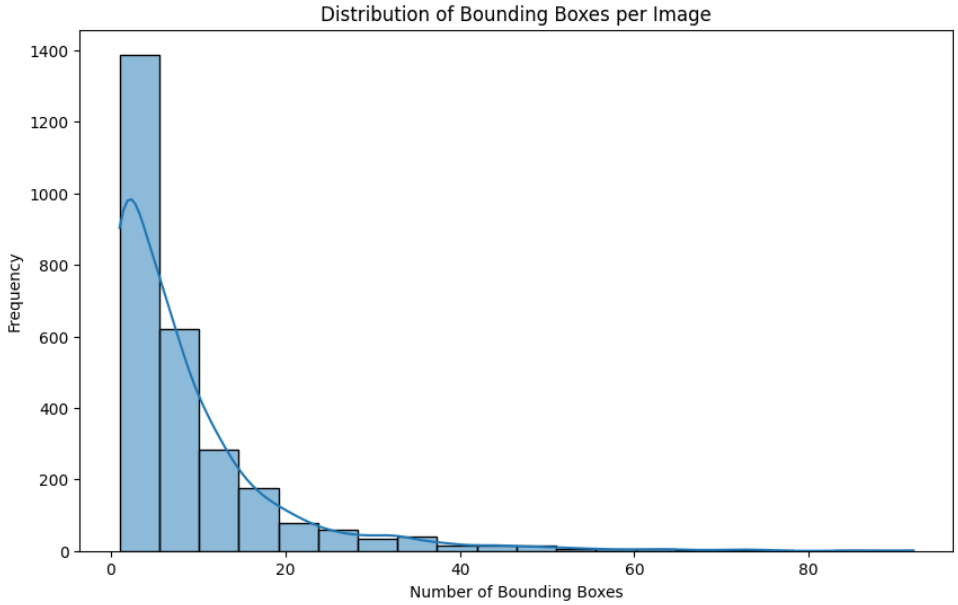


Figure 1: Distribution of bounding boxes per image

3.1.1 Class Distribution Across Bounding Boxes

Table 2 shows the distribution of each class across all bounding boxes. The majority of bounding boxes are labeled as *Trophozoite* (15,838), followed by *WBC* (7,004) and *NEG* (688).(also see Figure 3)

Class	Bounding Box Count
Trophozoite	15,838
WBC	7,004
NEG	688

Table 2: Class Distribution Across Bounding Boxes

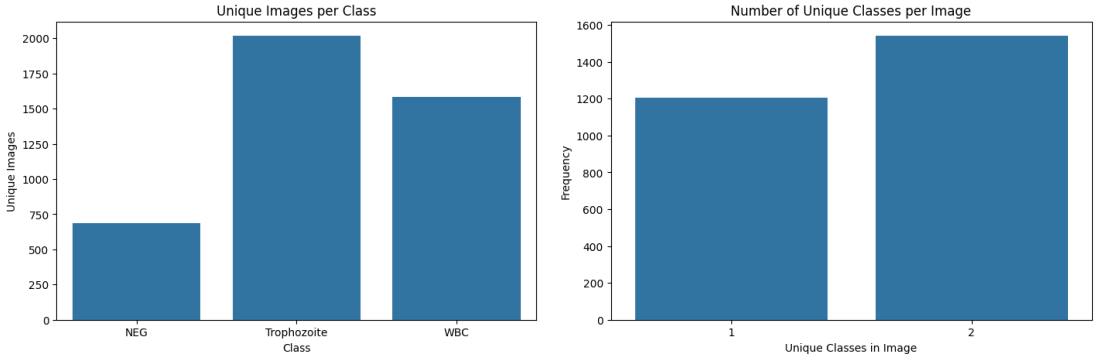
3.1.2 Class Occurrence per Unique Image

The occurrence of each class per unique image is summarized in Table 3. *Trophozoite* appears in 2,018 unique images with an average of 0.73 bounding boxes per image, while *WBC* and *NEG* appear in 1,583 and 688 unique images, respectively, with an average bounding box count of 0.58 and 0.25 per image.

Class	Unique Images	Average Bounding Boxes per Image
NEG	688	0.25
Trophozoite	2,018	0.73
WBC	1,583	0.58

Table 3: Class Occurrence per Unique Image

The dataset comprises 2,747 unique images. However, analysis of the class occurrence per unique image reveals that the total count of unique images in each class surpasses this initial figure, suggesting that some images contain instances of more than one class. Figure 2b provides a summary of unique class occurrences across images, showing that, nearly half of the images are annotated with multiple classes, potentially capturing more complex scenes.



(a) Number of unique images per class (b) Number of unique classes per image

Figure 2: Analysis of Image Class Distribution: (a) Number of unique images per class, and (b) Number of unique classes per image.

This analysis highlights that certain images are categorized into more than one class, which may require specialized preprocessing to handle these multi-class images effectively. Given the class imbalance and the presence of multi-class images, future model development may benefit from data balancing techniques and consideration of multi-class detection strategies.
The dataset exhibits a wide variance in bounding box counts per image, with some images containing multiple objects of interest. The *Trophozoite* class is the most frequently labeled, indicating a possible imbalance in class distribution. This analysis provides essential insights for preprocessing and model development, suggesting that data balancing techniques may be necessary to improve model performance.

3.2 Class Distribution and Statistical Summary

This visualization in Figure 3 shows the distribution of samples across the three classes in the dataset. The distribution reveals that the dataset is imbalanced, with a significantly higher number of samples for the Trophozoite class compared to the other classes.

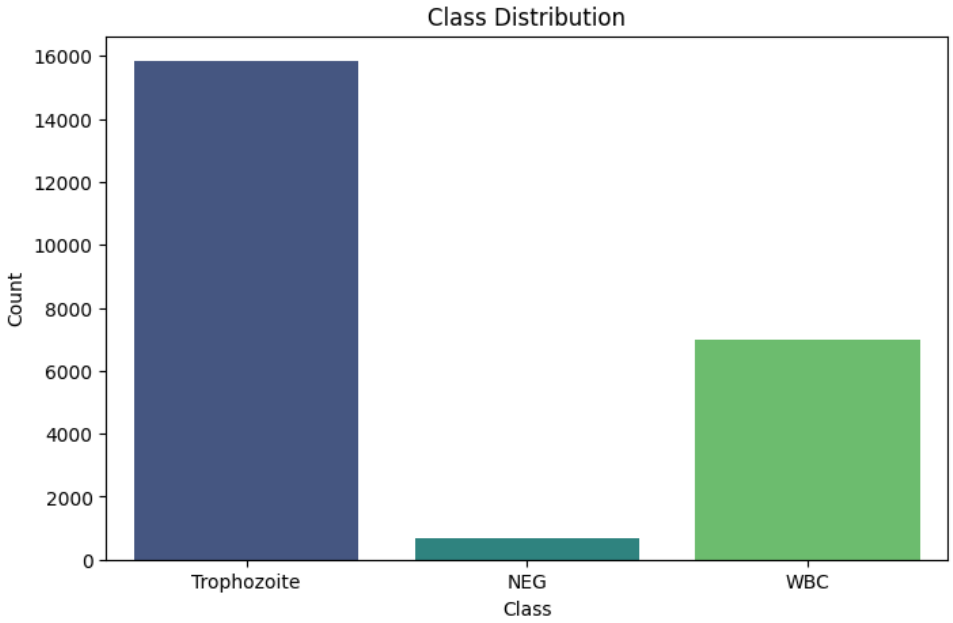


Figure 3: Count plot showing the distribution of samples across different classes in the dataset.

The dataset includes three classes with the following distribution:

- **Trophozoite**: 15,838 samples
- **WBC**: 7,004 samples
- **NEG**: 688 samples

3.2.1 Statistical Summary of Key Features

Table 4 provides a statistical summary for the confidence scores and bounding box coordinates (`ymin`, `xmin`, `ymax`, `xmax`) across the dataset. The boxplot in Figure 4 visualizes the distribution of bounding box coordinates in the dataset. The plot provides insights into the spread and potential outliers of the coordinate values, which can help assess the consistency and quality of the annotated bounding boxes for the images.

Statistic	`confidence`	`ymin`	`xmin`	`ymax`	`xmax`
Count	23,530	23,530	23,530	23,530	23,530
Mean	1.0	802.80	1292.34	878.26	1366.81
Standard Deviation	0.0	676.15	860.77	704.87	886.52
Minimum	1.0	0.0	0.0	0.0	0.0
25^{th} Percentile	1.0	317.0	685.0	373.0	740.0
50^{th} Percentile (Median)	1.0	619.0	1107.0	681.0	1169.0
75^{th} Percentile	1.0	1016.75	1621.0	1108.0	1701.0
Maximum	1.0	3012.0	4051.0	3116.0	4155.0

Table 4: Statistical Summary of Confidence Scores and Bounding Box Coordinates

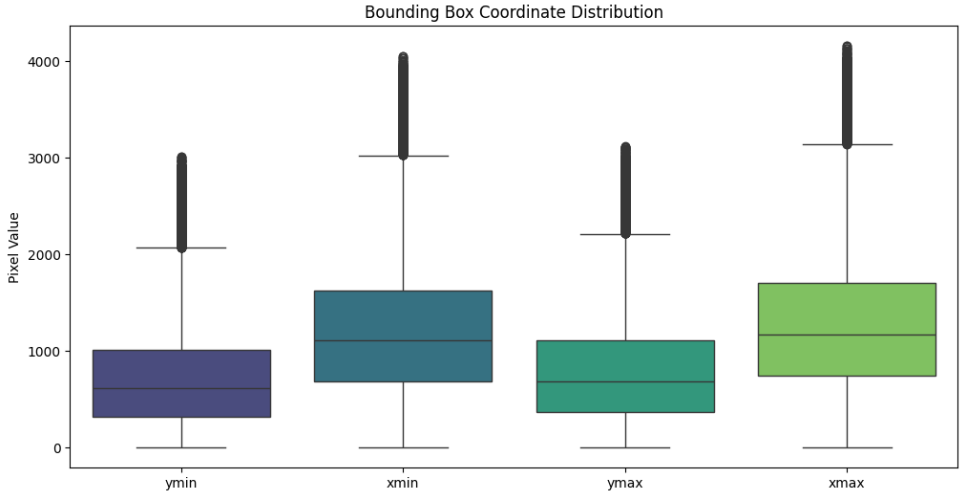


Figure 4: Distribution of bounding box coordinates across the dataset, highlighting variations in pixel values for `ymin`, `xmin`, `ymax`, and `xmax`.

3.3 Image Quality Analysis

The analysis of image datasets is crucial for understanding the quality and consistency of the data prior to applying machine learning models, especially in tasks involving convolutional neural networks (CNNs). This report examines the uniformity of image sizes and quality in the dataset, which contains images of varying dimensions.

First, to visually assess the dataset, we selected five random images per class, allowing us to observe key visual features and distinctions across classes (Figure 5).

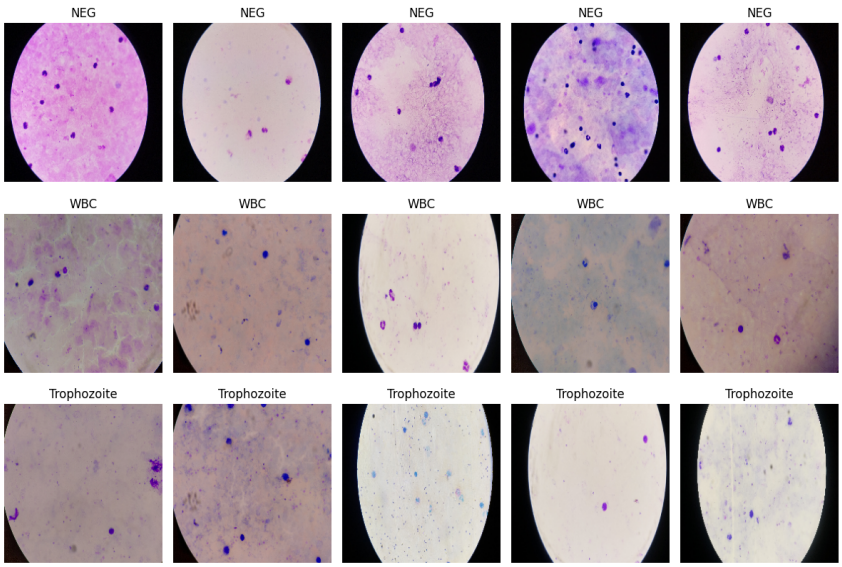


Figure 5: Sample images from each class in the blood sample dataset, illustrating visual differences between classes. The top image represents NEG (Negative for Malaria), the middle image shows WBC (White Blood Cells), and the bottom image depicts Trophozoite (Malaria parasite stage). These examples highlight distinct visual characteristics within each class, aiding in model training and classification

3.3.1 Brightness, Contrast, and Noise Level Analysis

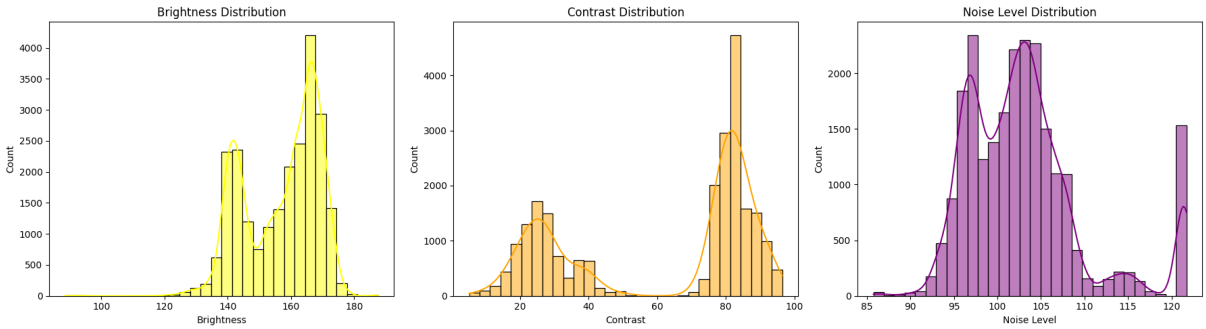


Figure 6: Distribution of Brightness, Contrast, and Noise Level

To analyze image quality, we measured three main aspects: *brightness, contrast, and noise level* across the dataset. *Brightness* was calculated as the mean pixel intensity of each image, which provides a general measure of the exposure level. Higher mean values indicate brighter images, while lower values indicate darker ones. *Contrast* was determined by calculating the standard deviation of pixel intensities within each image. This measure reflects the variation in intensity values, with higher standard deviations indicating images with more defined differences between light and dark areas. For *noise estimation*, we approximated noise level by measuring the standard deviation of the difference between the original image and a Gaussian-blurred version. This approach provides an indication of the variation in pixel intensities that are not associated with the actual content of the image, thus serving as a proxy for noise.
Distributions of brightness, contrast, and noise levels were plotted in Figure 6 to visualize the spread and uniformity across the images.

Statistic	**Brightness**	**Contrast**	**Noise Level**
Count	23530	23530	23530
Mean	157.05	61.88	103.002
Std	11.79	27.85	6.78
Min	88.51	5.19	85.75
25%	144.91	28.56	97.85
50% (Median)	160.78	78.72	102.25
75%	166.76	83.04	105.64
Max	187.59	96.47	121.72

Table 5: Summary Statistics for Brightness, Contrast, and Noise Level

- **Brightness**
  - The average brightness level is 157, which suggests that the images are generally well-lit. However, there may still be some variation due to the standard deviation (11.8).

- Brightness values span from approximately 88.5 to 187.6. With most values concentrated around the median (160.8), this indicates some images might be dimmer or overexposed.

- **Contrast**
  - The mean contrast is about 61.9, with a substantial standard deviation of 27.9. This high variation suggests that the images vary widely in contrast.
  - Contrast values range from 5.2 to 96.5, showing significant diversity across images. A low contrast (close to 5) might make certain images less clear for model training.
  - A quarter of the images have contrast below 28.6, indicating a subset of images with low contrast. (These images will be deleted in preprocessing step) The median contrast is much higher (78.7), implying most images do have moderate to high contrast, which could be beneficial for object detection.

- **Noise Level**
  - With an average noise level around 103, the dataset has relatively consistent noise levels.
  - Noise levels are fairly close (from around 85.8 to 121.7), with a low standard deviation of 6.8. This suggests the noise level is fairly uniform across images.

3.3.2 Image Resolution Consistency Check

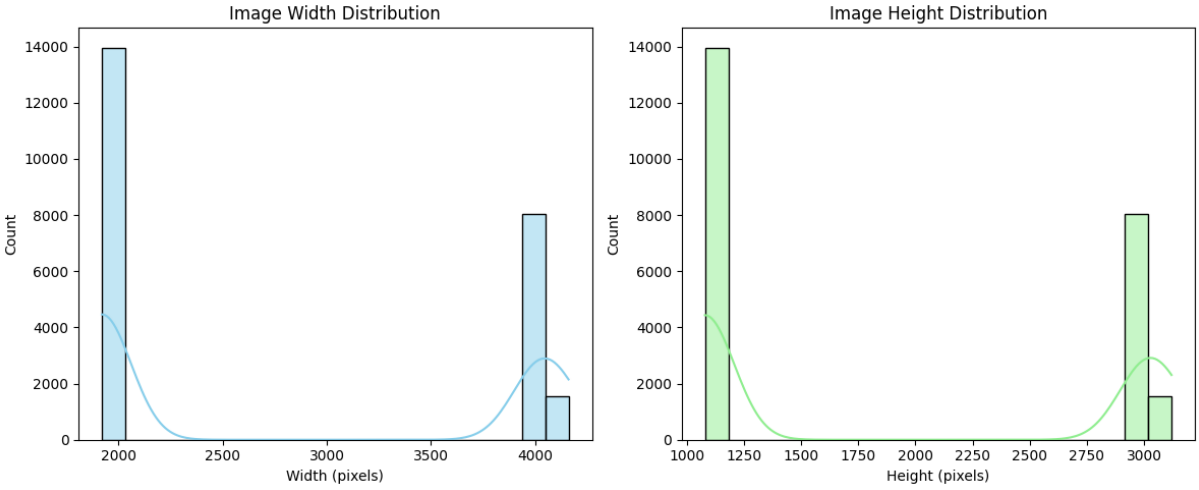


Figure 7: Distribution of Image Widths and Heights in Pixels. The left histogram illustrates the width distribution, while the right histogram depicts the height distribution, providing insights into the dimensional characteristics of the images within the dataset.

To evaluate the uniformity of image dimensions across the dataset, we analyzed the width and height of each image. For each sample, the width and height were extracted, allowing us to assess whether images had consistent dimensions or varied in size. This step is

crucial as CNNs require input images of a fixed resolution. Variations in image size could affect model performance and necessitate preprocessing steps like resizing or padding to achieve uniformity. We plotted the distributions of both width and height in Figure 7 to depict their spread and identify any significant deviations from a standard resolution. Additionally, we identified unique resolutions within the dataset in table 6. According to the table, the resolution sizes vary, indicating that resizing will be necessary during preprocessing to ensure all images meet the model's input requirements.

Width	Height	Count
1920	1080	13963
4000	3000	688
4032	3016	7344
4160	3120	1535

Table 6: Unique Image Resolutions and Their Counts

4 Preprocessing

4.1 Deleting Images Associated with More Than One Class

In this step, we identify and remove images that are associated with more than one class. The goal is to ensure that each image belongs to only one class, which is important for the consistency and reliability of our model.

To achieve this, we first group the data by the 'Image_ID' and count the number of unique classes associated with each image (Figure 2b). Any image associated with more than one class is removed. After applying this filtering step, we obtain a dataset with **3349** image.

4.2 Resizing Images

In order to improve the time efficiency of the data processing pipeline, we performed image resizing before applying data augmentation techniques. Resizing the images to a consistent target size ensures that subsequent steps, including augmentation, are computationally less expensive.

This resizing process ensures that each image is resized to `224 x 224` pixels. Additionally, the bounding box coordinates are adjusted proportionally to the resizing factor.

4.3 Image Quality Enhancement

In this step, we focused on enhancing the quality of the images to improve the performance of the model by removing low-contrast images and applying image processing techniques such as histogram equalization and Gaussian filtering.

- **Removal of Low Contrast Images:** To ensure that the dataset only contains images with adequate visual information, we removed images below a threshold of `28.6` (according to table 5).

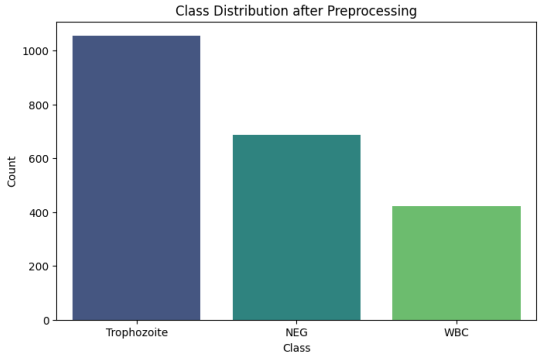
- **Histogram Equalization:** We applied histogram equalization to enhance the contrast and brightness. Histogram equalization spreads out the most frequent intensity values across the entire range, improving the visibility of details, especially in images with poor contrast.
- **Gaussian Filtering:** To further improve the image quality, Gaussian filtering was applied to reduce noise while preserving edges.

The total number of images was reduced to `2164` after applying these quality enhancements.

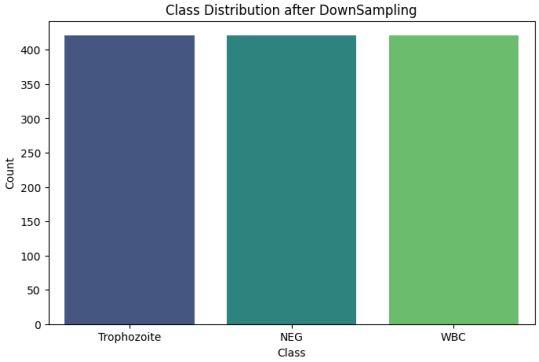
4.4 Balancing the Dataset

After preprocessing steps in the previous subsections, the dataset contained an imbalance in class distribution. To address this, we first visualized the class distribution as shown in Figure 8a.

To balance the dataset, we reduced the samples of the majority classes (`Trophozoite` and `NEG`) to match the size of the minority class (`WBC`, with 421 samples). This downsampling was done randomly to ensure each class had an equal number of samples, as shown in Figure 8b.



(a) Class distribution after preprocessing. The class imbalances are clearly visible, with `Trophozoite` having the highest count and `WBC` the lowest.



(b) Class distribution after downsampling. The dataset is now balanced, with each class containing 421 samples.

After balancing, the dataset was reset to ensure consistent indexing, and the resulting balanced dataset was used for further analysis and model training.

4.5 Data Augmentation

In this section, we apply various augmentations to enhance the robustness of the image data. The implemented augmentation methods introduce diverse transformations while ensuring the integrity of the regions defined by the corresponding bounding boxes.
We performed resizing to 224 x 224 pixels before applying data augmentation to improve time efficiency during preprocessing. However, certain augmentation techniques, such as random cropping, modify the image dimensions, requiring an additional resizing step.

Therefore, after applying random cropping, we resize the images again to 224 x 224 pixels to maintain a consistent input size for subsequent processing and model training.
The following augmentation techniques were applied to the original image:

- **Rotation 45°**
- **Horizontal Flip**
- **Vertical Flip**
- **Random Crop**:
  - Random Cropping technique was implemented to increase the variability of the training dataset by generating modified views of the images while preserving the relevant content within the bounding boxes.
  - We introduced small random variations to the crop coordinates by setting a range based on 10% of the image width and height.
- **Brightness Adjustment**
- **Contrast Adjustment**

For each augmentation technique, the new bounding box coordinates are recalculated and displayed. Figure 9 shows augmented versions of a sample image before preprocessing, while Figure 10 illustrates the augmented versions of a sample image after preprocessing.

Image	Augmentation Type	Bounding Box
Before Preprocessing Image_ID: id_a6cl90trri.jpg	Original	(1566, 558, 1604, 600)
	Rotation 45°	(1401.23, 97.35, 1457.80, 153.92)
	Horizontal Flip	(316, 558, 354, 600)
	Vertical Flip	(1566, 480, 1604, 522)
	Random Crop	(1549, 485, 1587, 527)
	Brightness Adjustment	(1566, 558, 1604, 600)
	Contrast Adjustment	(1566, 558, 1604, 600)
After Preprocessing Image_ID: id_n9alv46e3p.jpg	Original	(43, 25, 50, 33)
	Rotation 45°	(1.69, 94.32, 12.30, 104.93)
	Horizontal Flip	(174, 25, 181, 33)
	Vertical Flip	(43, 191, 50, 199)
	Random Crop	(28.0, 4.0, 35.0, 12.0)
	Brightness Adjustment	(43, 25, 50, 33)
	Contrast Adjustment	(43, 25, 50, 33)

Table 7: Bounding Box Coordinates for Augmented Versions of Two Different Sample Images, One Before Preprocessing and One After Preprocessing. The table shows the bounding box coordinates (xmin, ymin, xmax, ymax) for these images undergoing various augmentation techniques.

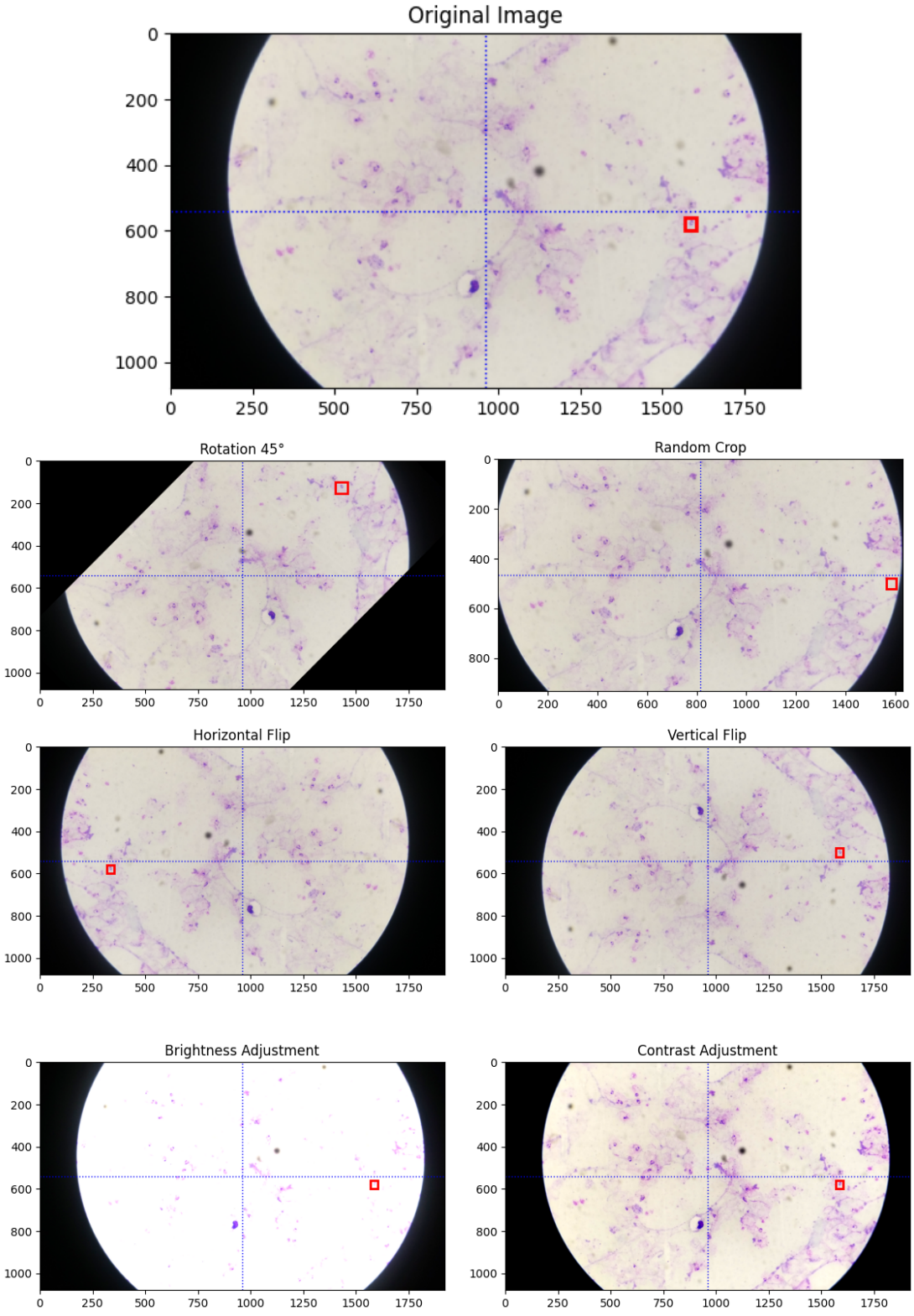


Figure 9: Image Augmentation Techniques Applied on Image_ID: id_a6cl90trri.jpg: Original Image, Rotated Image, Horizontal Flip, Vertical Flip, Random Cropped Image, Brightness Adjustment, and Contrast Adjustment.

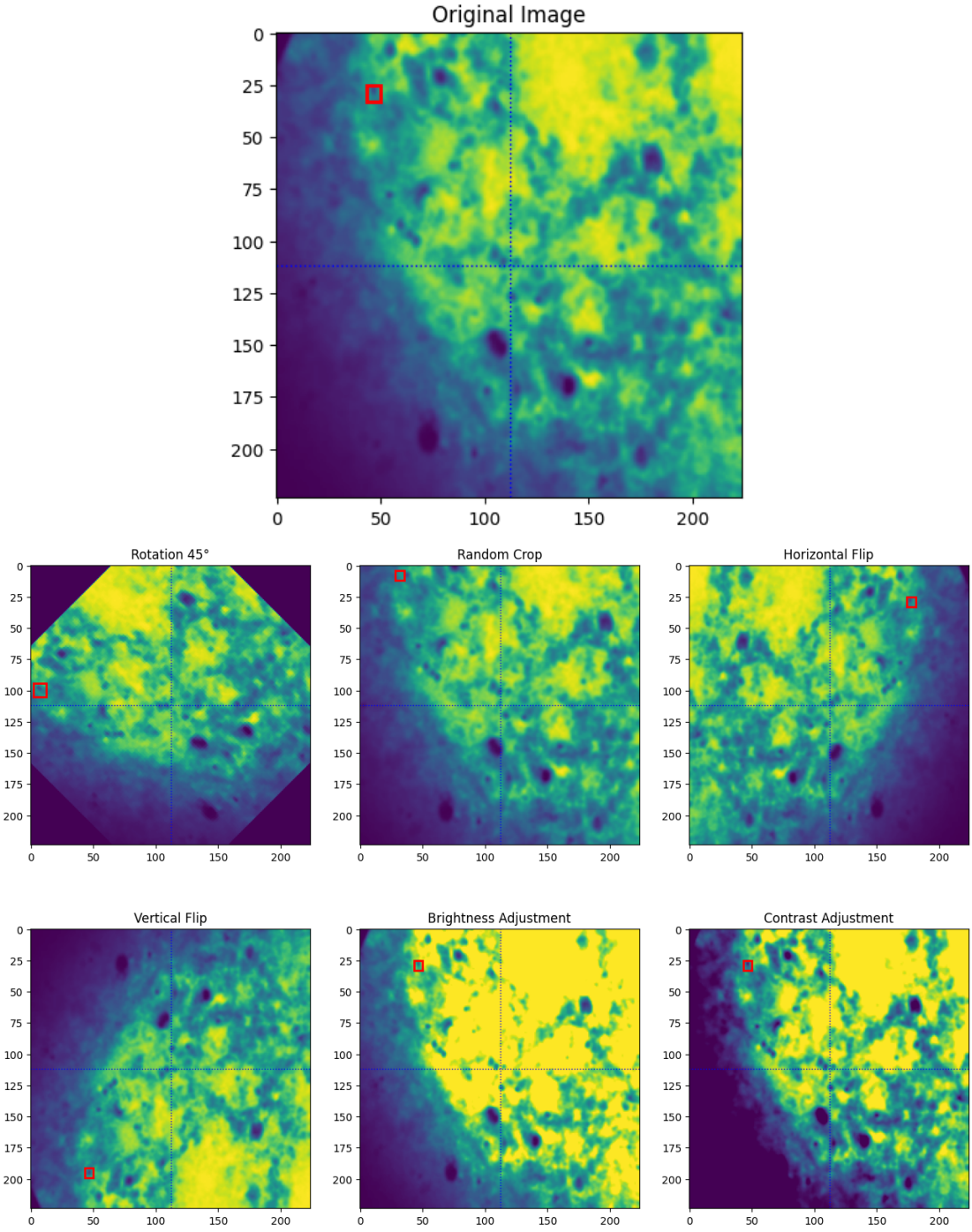


Figure 10: Image Augmentation Techniques Applied on Image_ID: id_n9alv46e3p.jpg: Original Image, Rotated Image, Horizontal Flip, Vertical Flip, Random Cropped Image, Brightness Adjustment, and Contrast Adjustment.

We apply the augmentation techniques described above to artificially increase the number of samples for each to 1000 samples.

4.6 Dataset Creation and Splitting

In this section, we describe the process of transforming the image data and splitting it into training, validation, and test sets. The dataset consists of images and associated bounding box annotations, which are processed and organized into a custom dataset class.

The transformation applied to the images includes converting them into tensors and **normalizing** the pixel values using the ImageNet mean and standard deviation values. The bounding box coordinates are also normalized to ensure they are relative to the image dimensions.

The dataset is divided into three subsets: training, validation, and testing. The training set consists of 1920 samples, the validation set contains 480 samples, and the test set has 600 samples. The data is loaded using custom DataLoader objects to ensure efficient batching during training and evaluation.

5 Model Design and Hyper Parameter Tuning

5.1 Model Design

For our classification and bounding box regression tasks, we designed a Convolutional Neural Network (CNN) architecture. The model comprises two main components: feature extraction layers and classification/regression heads.

The feature extraction layers consist of a series of convolutional layers, each followed by ReLU activation and max-pooling. The architecture includes:

- Convolutional Layers: We start with three channels (RGB) in the input, followed by sequential layers that increase the number of feature maps: 32, 64, 128, and 256.
- Activation Functions: ReLU is used to introduce non-linearity after each convolutional layer.
- Pooling Layers: Max-pooling is applied after each convolutional layer to reduce spatial dimensions while retaining the most salient features.

After feature extraction, we have two distinct heads:

- **Classification Head**: This includes fully connected layers that output the probabilities for the three classes: NEG, Trophozoite, and WBC. It employs dropout for regularization.
- **Bounding Box Regression Head**: This head predicts the bounding box coordinates, consisting of four outputs: $xmin$, $ymin$, $xmax$, and $ymax$.

The forward method processes the input through the feature layers and then splits the output for classification and regression. The model architecture used for this classification task is a convolutional neural network (CNN) with several convolutional layers followed by fully connected layers. The layers are designed to extract features from the input images, followed by classification layers for the final output.

5.2 Model Architecture

The detailed architecture of the network is shown in table 8, with the output shape and parameter count for each layer:

Layer (type)	**Output Shape**	**Param #**
Conv2d-1	[-1, 32, 224, 224]	896
ReLU-2	[-1, 32, 224, 224]	0
MaxPool2d-3	[-1, 32, 112, 112]	0
Conv2d-4	[-1, 64, 112, 112]	18,496
ReLU-5	[-1, 64, 112, 112]	0
MaxPool2d-6	[-1, 64, 56, 56]	0
Conv2d-7	[-1, 128, 56, 56]	73,856
ReLU-8	[-1, 128, 56, 56]	0
MaxPool2d-9	[-1, 128, 28, 28]	0
Conv2d-10	[-1, 256, 28, 28]	295,168
ReLU-11	[-1, 256, 28, 28]	0
MaxPool2d-12	[-1, 256, 14, 14]	0
Linear-13	[-1, 512]	25,690,624
ReLU-14	[-1, 512]	0
Dropout-15	[-1, 512]	0
Linear-16	[-1, 3]	1,539
Linear-17	[-1, 512]	25,690,624
ReLU-18	[-1, 512]	0
Dropout-19	[-1, 512]	0
Linear-20	[-1, 4]	2,052
Total Params		51,773,255
Trainable Params		51,773,255
Non-Trainable Params		0

Table 8: Model architecture and parameter summary.

5.3 Hyper Parameter Tuning

To optimize model performance, we carefully tuned several hyperparameters during the training process. Key hyperparameters include:

- **Learning Rate**: We experimented with different learning rates to find the optimal value that promotes rapid convergence without overshooting the minima.
- **Batch Size**: The batch size of 32 was selected based on memory constraints and the trade-off between training time and stability.

- **Dropout Rate**: A dropout rate of 0.5 was used in the classification and regression heads to mitigate overfitting.
- **Number of Epochs**: We monitored performance metrics to determine the appropriate number of epochs for training, ensuring that we achieved a balance between underfitting and overfitting.

6 Model Training and Validation

In this section, we outline the training and validation processes for our model.

6.1 Training Process

The training process was conducted using the **Adam optimizer** with a learning rate of 0.001, allowing for adaptive learning. We defined two loss functions:

- **Classification Loss**: CrossEntropyLoss was employed to evaluate the performance of the classification head.
- **Bounding Box Loss**: SmoothL1Loss was used for the regression head to measure the accuracy of the predicted bounding box coordinates.

The training routine involved iterating through the training dataset for a specified 5 number of epochs. During each epoch:

- The model was set to training mode, and the total classification and bounding box losses were initialized.
- For each batch of images, class labels, and bounding boxes, the following steps were performed:
  - Gradients were reset before the forward pass.
  - The model produced outputs for class predictions and bounding boxes.
  - The classification and bounding box losses were computed, and the total loss was derived by summing the two.
  - Backpropagation was applied to update the model parameters through the optimizer.
  - The cumulative losses for classification and bounding box regression were recorded for reporting.
- At the end of each epoch, the model's performance was validated using a separate validation dataset.

6.2 Validation Process

The validation process was performed after each training epoch to assess the model's generalization ability. The validation routine included:

- Setting the model to evaluation mode to disable dropout layers and batch normalization effects.
- Calculating the classification and bounding box losses over the validation dataset without updating the model parameters (using no gradient computation).

The validation losses were averaged over the entire validation dataset to provide a clear indication of model performance.

7 Results

7.1 Loss

The model was trained for 5 epochs with both classification and bounding box loss evaluated at each epoch. The training and validation losses progressively decreased, demonstrating convergence in both tasks.

Epoch	Train Classification Loss	Train BBox Loss	Val Classification Loss	Val BBox Loss
1	0.63	0.11	0.20	0.04
2	0.12	0.05	0.07	0.04
3	0.03	0.04	0.03	0.04
4	0.01	0.04	0.02	0.04
5	0.01	0.04	0.01	0.04
6	0.02	0.04	0.02	0.04
7	0.01	0.04	0.02	0.04
8	0.00	0.04	0.03	0.04
9	0.02	0.04	0.03	0.04
10	0.02	0.04	0.00	0.04

Table 9: Training and Validation Losses for Classification and Bounding Box Tasks

7.2 Confusion Matrix

The confusion matrix for the classification task is shown in Figure 11.

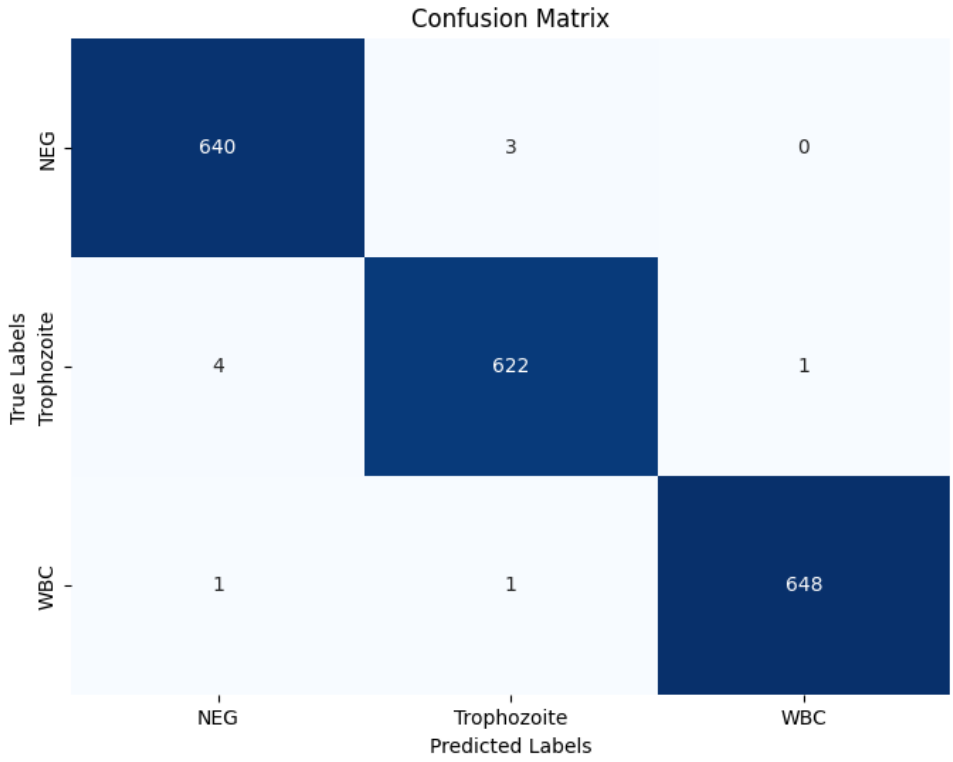


Figure 11: Confusion Matrix for Classification Task

The confusion matrix reflects the model's classification accuracy across different classes. The diagonal elements (640 for Class NEG, 622 for Class Trophozoite, and 648 for Class WBC) represent the correctly classified instances for each class. The high values along the diagonal indicate that the model is performing well, with very few misclassifications across the classes.