# Analysis of the BloodMNIST Dataset Using Color Histograms and Random Forest

Yasharzadeh M., Sokolovsky A., Yosupov L.

## 1. Abstract

In this work, we analyze the BloodMNIST dataset, which comprises images of blood cells divided into eight categories. We initially implement methods for extracting color histograms from each image, creating feature vectors that describe color distributions. We then merge the training (train) and validation (val) sets to train a Random Forest model. We evaluated the model on the test set using multiple metrics: Accuracy, AUC, a confusion matrix and a classification report. The results indicate an overall Accuracy of about 89% and an AUC of about 0.9888, signifying good separation among most categories. This paper outlines the model-building and analysis steps and suggests directions for future improvement.

## 2. Introduction

Rapid progress in the fields of Computer Vision and Machine Learning has provided numerous solutions for analyzing medical images. Among these tasks, the classification of blood cells plays a critical role in diagnosing hematological disorders and detecting parasites (e.g., malaria). The BloodMNIST dataset, part of the MedMNIST project[1], is designed to facilitate quick experimentation in blood cell detection and classification, featuring small images (28×28 pixels) and a straightforward division into training, validation, and test sets.

In this work, we apply a classic color histogram extraction technique for each image, resulting in feature vectors that capture the color distribution. We then chose Random Forest as our classification algorithm, known for its robustness and efficiency. The following sections detail our methodology and present the results obtained on the test set.

## 3. Literature Review

### 3.1. Significance of Blood Cell Classification and BloodMNIST

Automatic blood cell classification has garnered considerable research attention, as it can save time and reduce human error in medical labs[2]. Different cell types are characterized by features such as color, shape, and size.

- **BloodMNIST**: Provided as part of the MedMNIST project[1], it offers real samples of blood cells in a small format, serving as an initial benchmark for Machine Learning or Deep Learning applications in the medical imaging domain.
- **Use Cases**: Identifying abnormal blood cells can indicate various diseases, such as malaria, anemia, and leukemia, as well as inflammatory conditions.

### 3.2. Color Histogram Extraction

In classic computer vision, color histograms are a widely used technique for describing color distribution in images. Color histograms offer a compact representation that is relatively invariant to small translations of pixels but do not capture spatial information, which can lead to loss of morphological details.

### 3.3. Random Forest Algorithm

Random Forest[3] is an ensemble method based on multiple decision trees whose predictions are averaged. It is resilient to noise, handles heterogeneous data well, and provides interpretability through feature importance measures.

### 3.4. Deep Models and Other Approaches

Convolutional Neural Networks (CNNs) have become a central tool for image recognition tasks, including microscopic images of blood cells. Networks such as ResNet, VGG, and Inception often achieve impressive results but require large datasets and significant computational power.

## 4. Methodology

### 4.1. Dataset Description

MedMNIST, a large-scale MNIST-like collection of standardized biomedical images, including 12 datasets for 2D and 6 datasets for 3D. All images are standardized into multiple size options (MNIST-like 28 and larger 64/128/224) with the corresponding classification labels, so that no background knowledge is required for users. Each sub-dataset is pre-processed into the same format- BloodMNIST : 28*28 RGB, Divided in the 3 datasets:

- Train set: 11959 samples
- Validation set: 1712 samples
- Test set: 3421 samples
- Total dataset: 17092 samples

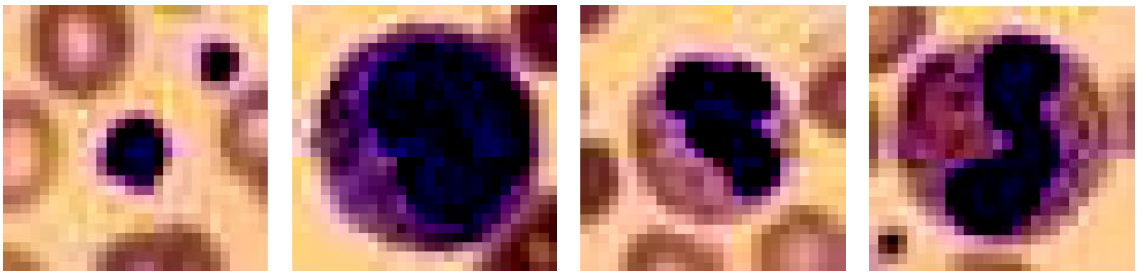

*Figure 1- Examples of Data Pictures*

### 4.2. Constructing the Feature Vector (Color Histogram)

Each image is represented by a set of color histograms - one per color channel (R, G, B) - where the pixel intensity range is divided into multiple bins. Each channel's histogram contributes multiple features, forming a feature vector for classification. Feature extraction using color histograms. The image array is transposed from shape (C, H, W) (channels, height, width) to (H, W, C) (height, width, channels) so that each color channel can be processed individually. For each channel (red, green, blue), a histogram is computed with a specified number of bins (default 16). The histogram counts the number of pixels falling into each bin, which helps capture the distribution of color intensities. The histograms from all channels are concatenated into a single feature vector.

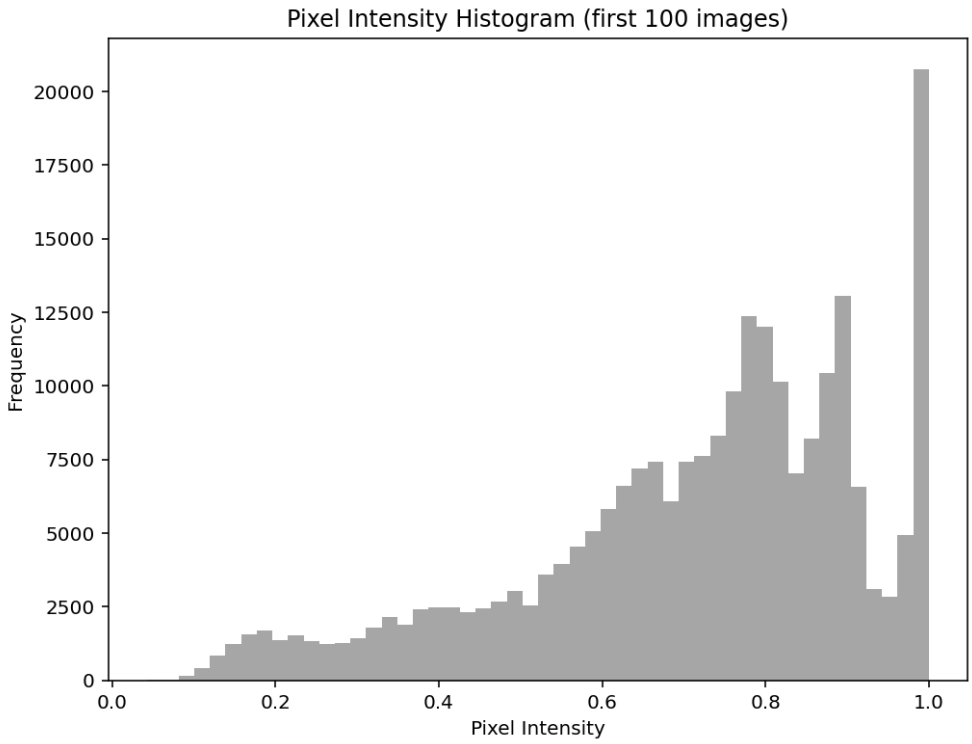


***Figure 2- Pixel Intensity Histogram (for first 100 images)***

### 4.3. Training the Random Forest Model

A Random Forest classifier is trained on the color histogram feature vectors, and predictions are made on the test set. Metrics such as Accuracy, AUC, and the confusion matrix are used for evaluation.

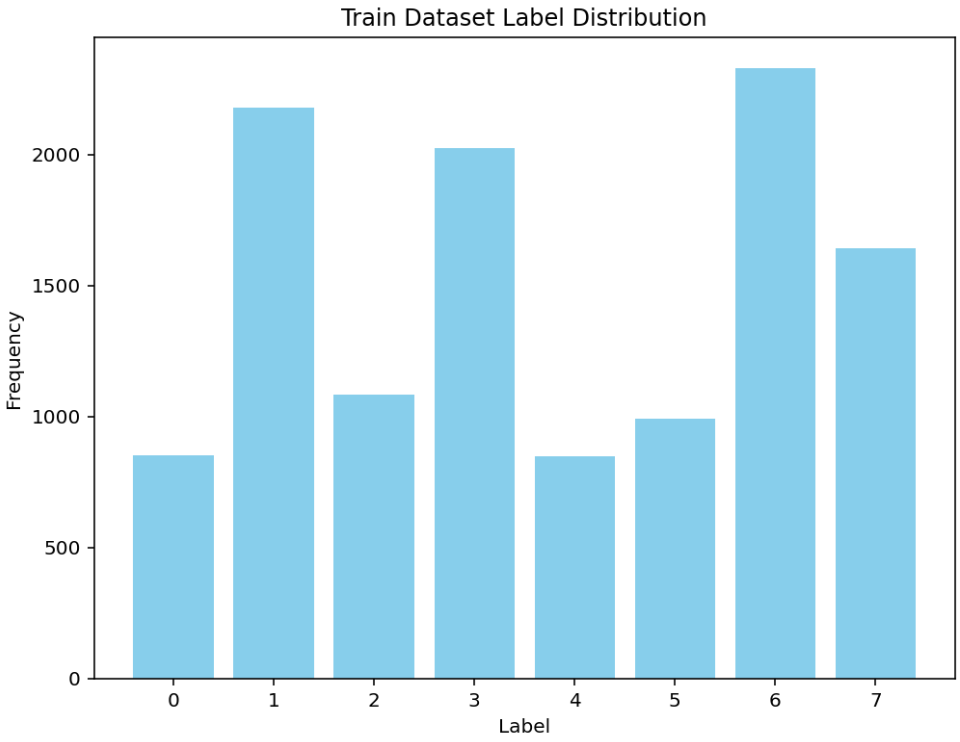


***Figure 3- Train Dataset Label Distribution***

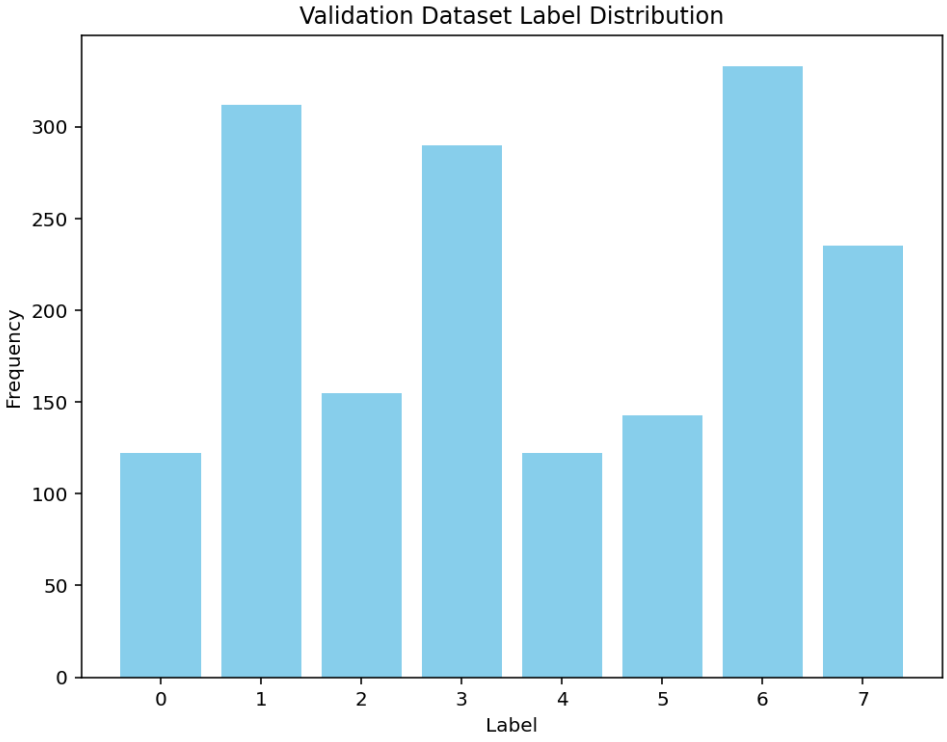




***Figure 4- Validation Dataset Label Distribution***

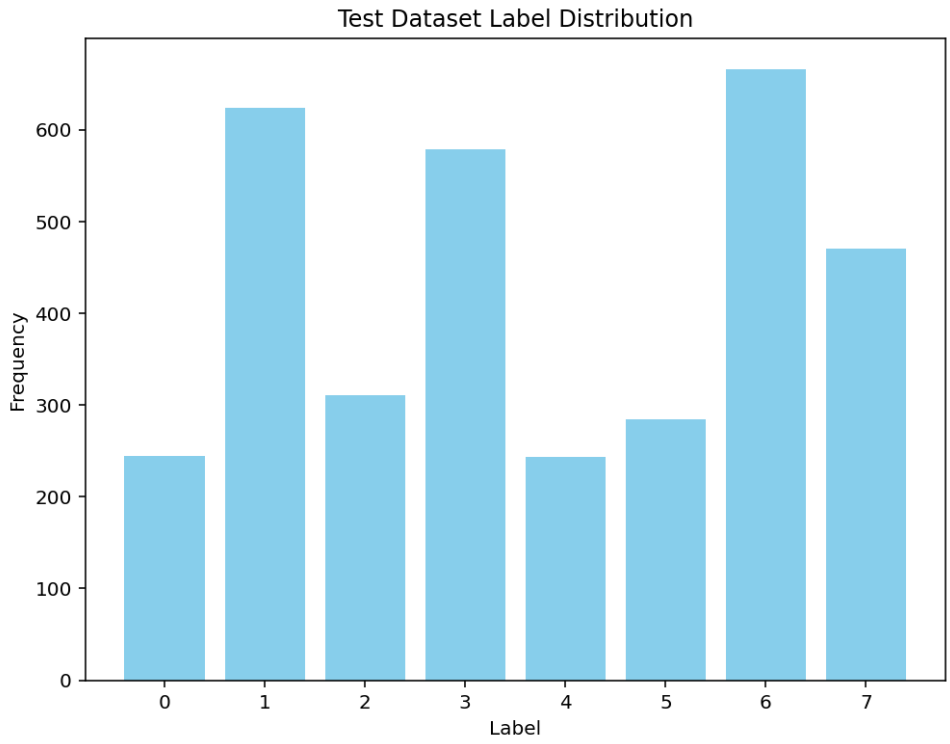


***Figure 5- Test Dataset Label Distribution***

## 5. Results

Here we will show the diffrences between thhe used methods' and the impact of hyperparmeters changing.

Classification Report: The model achieves an overall accuracy of ~89%. Some categories exhibit near-perfect Precision and Recall.

ROC-AUC: Approximately 0.9888, indicating strong class separation.

Confusion Matrix: Some confusion appears between classes 0 and 3, or 3 and 6, but others are almost perfectly identified.

ROC Curves: Most classes show curves near the upper-left corner (low FPR, high TPR), aligning with a high AUC.

| class | Precision | Recall | F1-Score | Support |
|---|---|---|---|---|
| 0 | 0.79 | 0.79 | 0.79 | 244 |
| 1 | 0.98 | 0.98 | 0.98 | 624 |
| 2 | 0.96 | 0.9 | 0.93 | 311 |
| 3 | 0.74 | 0.81 | 0.77 | 579 |
| 4 | 0.95 | 0.86 | 0.9 | 243 |
| 5 | 0.87 | 0.74 | 0.8 | 284 |
| 6 | 0.9 | 0.93 | 0.92 | 666 |
| 7 | 1 | 1 | 1 | 470 |
| Accuracy | 0.89 | | 3421 | |
| Macro Avg. | 0.9 | 0.88 | 0.89 | 3421 |
| Weighted Avg. | 0.9 | 0.89 | 0.9 | 3421 |

***Table 1 – Random Forest Model Training Results***

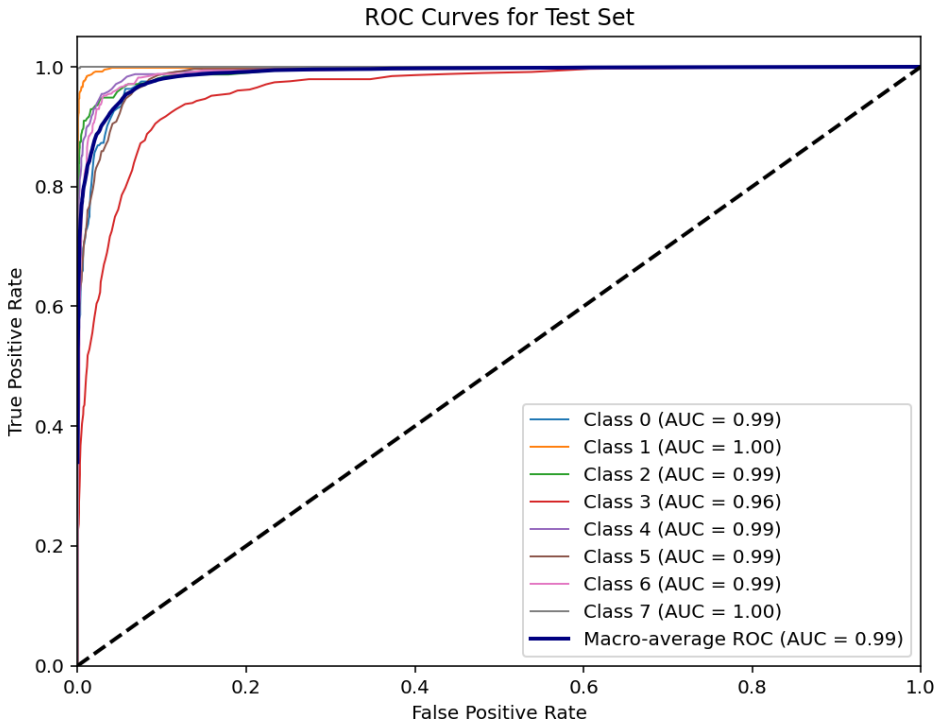


***Figure 6- Random Forest Model Training Results ROC***

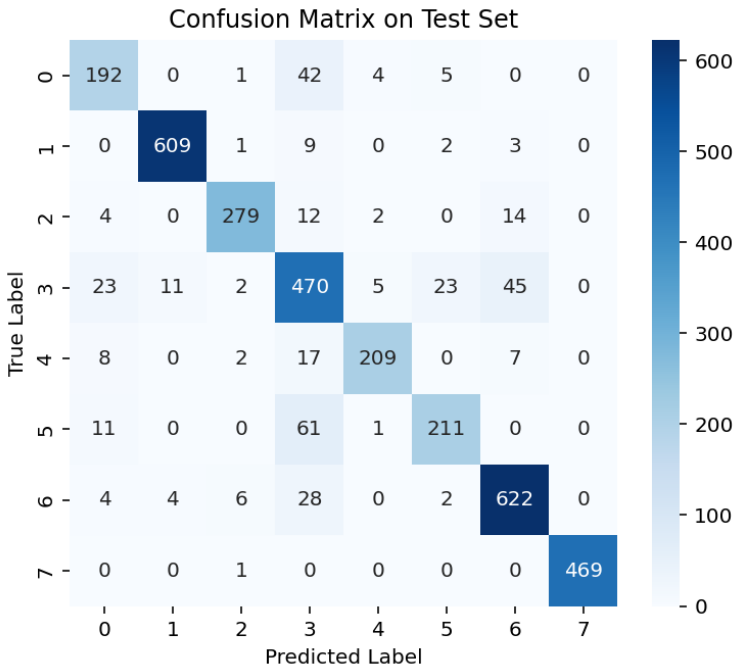


***Figure 7- Random Forest Model Training Confusion Matrix***

Now, we will use simple CNN algorithm:

| class | Precision | Recall | F1-Score | Support |
|---|---|---|---|---|
| 0 | 0.77 | 0.75 | 0.76 | 244 |
| 1 | 0.96 | 0.97 | 0.97 | 624 |
| 2 | 0.96 | 0.85 | 0.9 | 311 |
| 3 | 0.66 | 0.83 | 0.73 | 579 |
| 4 | 0.96 | 0.65 | 0.77 | 243 |
| 5 | 0.84 | 0.6 | 0.7 | 284 |
| 6 | 0.9 | 0.96 | 0.93 | 666 |
| 7 | 0.99 | 0.99 | 0.99 | 470 |
| Accuracy | 0.87 | | 3421 | |
| Macro Avg. | 0.88 | 0.83 | 0.84 | 3421 |
| Weighted Avg. | 0.88 | 0.87 | 0.87 | 3421 |

***Table 2 – CNN Results***

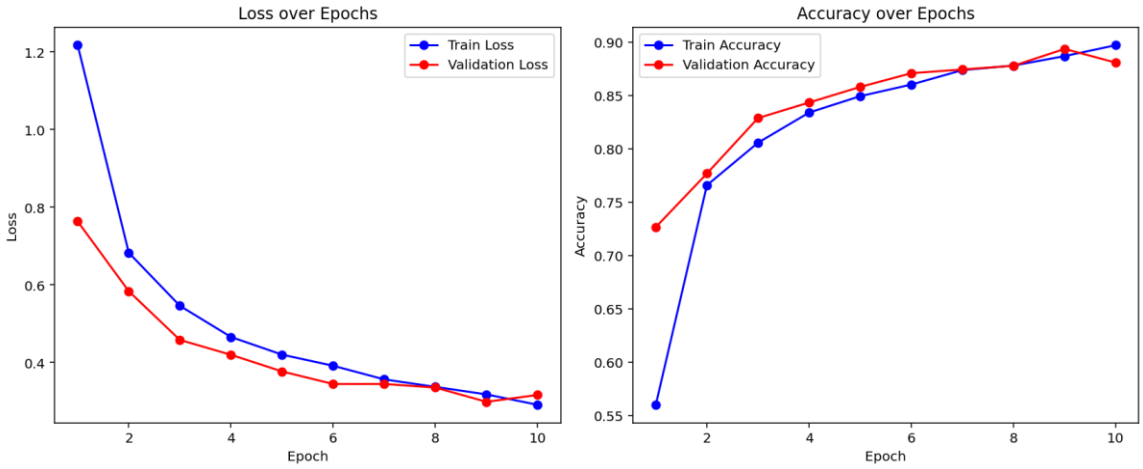


***Figure 8 – Epochs Impact on Lost & Accuracy***

On the next step, we going to activate neuron network with batch size of 64 and Sigmoiod function optimizer with (lr=0.001, momentum=0.9):

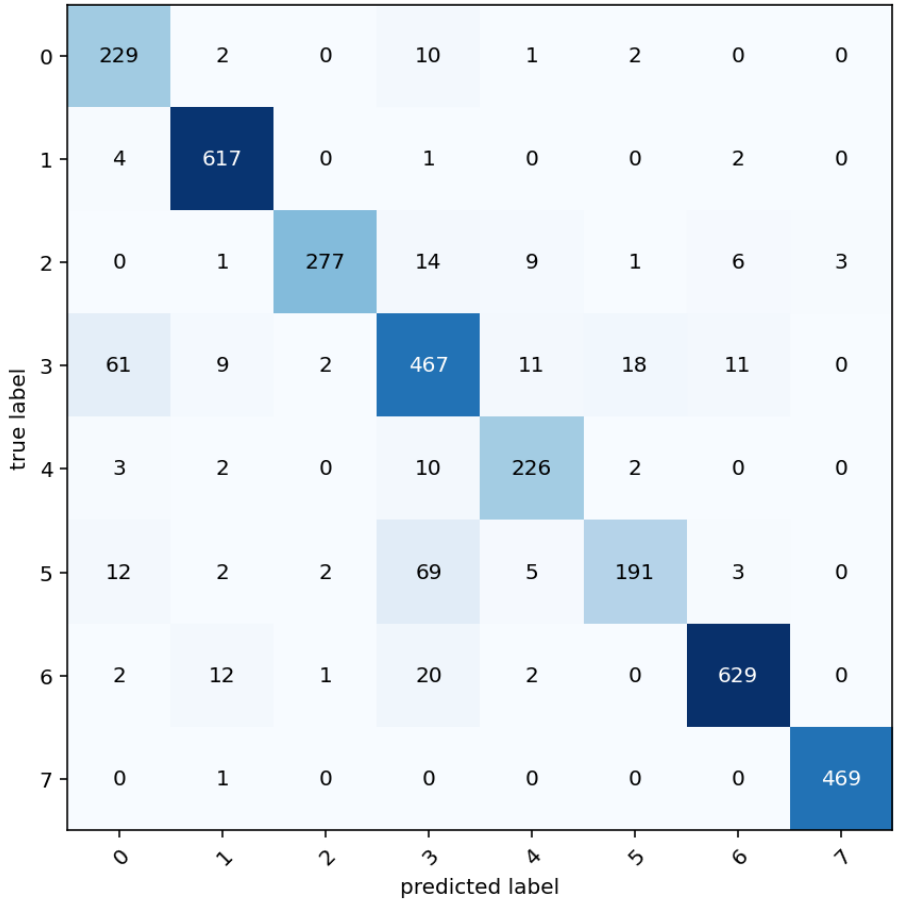


***Figure 9- Confusion Matrix, Sigmoid Optimizer, BS=64, lr=0.001, Momentum = 0.9***

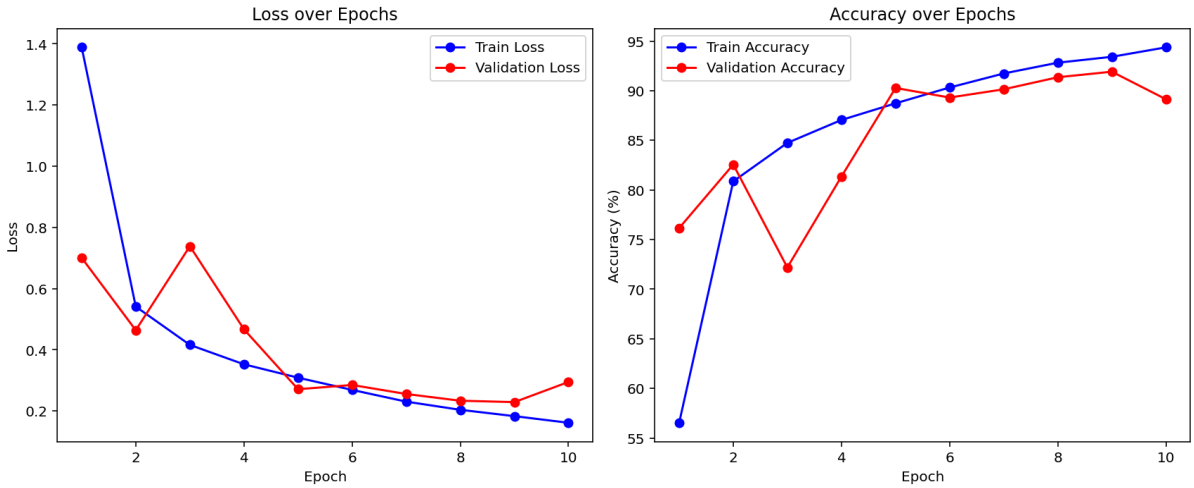


***Figure 10 – Epochs Impact on Lost & Accuracy***

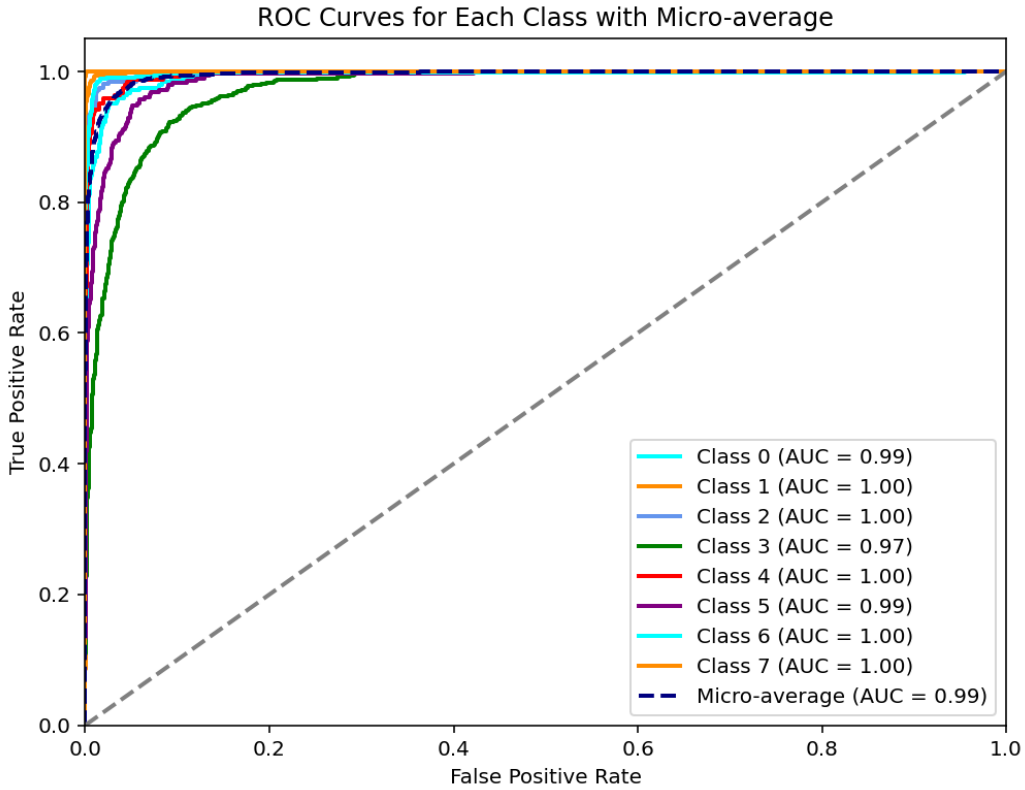


***Figure 11- ROC, BS=64, lr=0.001, Momentum = 0.9***

| class | Precision | Recall | F1-Score | Support |
|---|---|---|---|---|
| 0 | 0.907629 | 0.736334 | 0.938525 | 0.825225 |
| 1 | 0.907629 | 0.955108 | 0.988782 | 0.971654 |
| 2 | 0.907629 | 0.982270 | 0.890675 | 0.934233 |
| 3 | 0.907629 | 0.790186 | 0.806563 | 0.798291 |
| 4 | 0.907629 | 0.889764 | 0.930041 | 0.909457 |
| 5 | 0.907629 | 0.892523 | 0.672535 | 0.767068 |
| 6 | 0.907629 | 0.966206 | 0.944444 | 0.955201 |
| 7 | 0.907629 | 0.993644 | 0.997872 | 0.995754 |

***Table 3 – Sigmoid Optimizer Result Table***

Now, we change the batch size to 32 with adam optimizer ewith learning rate of 0.005:

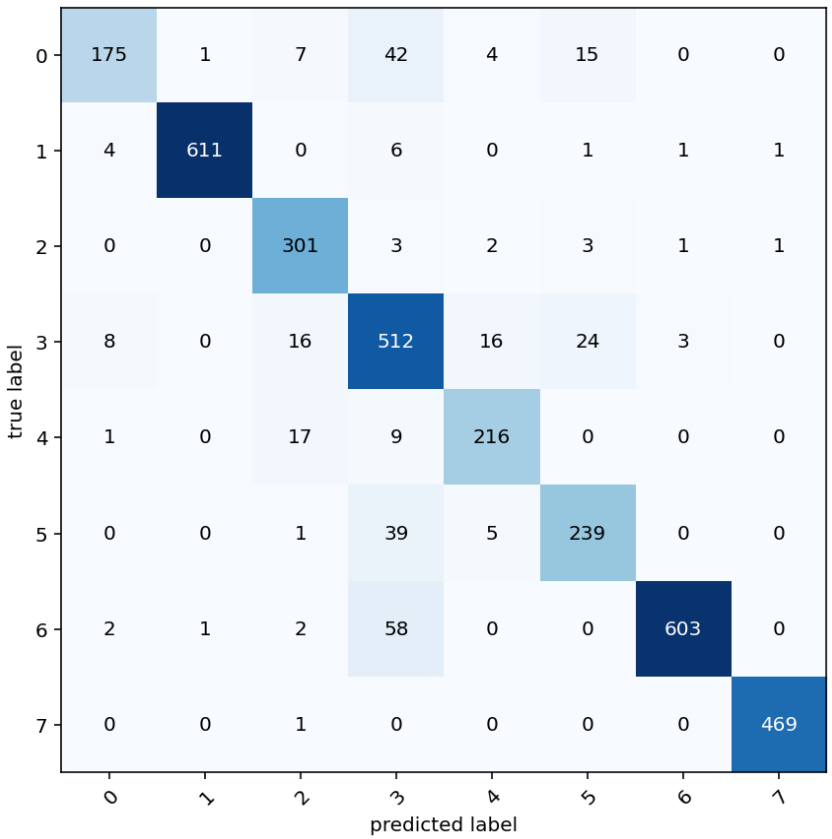

***Figure 12- Confusion Matrix, Adam Optimizer, BS=32, lr=0.005***

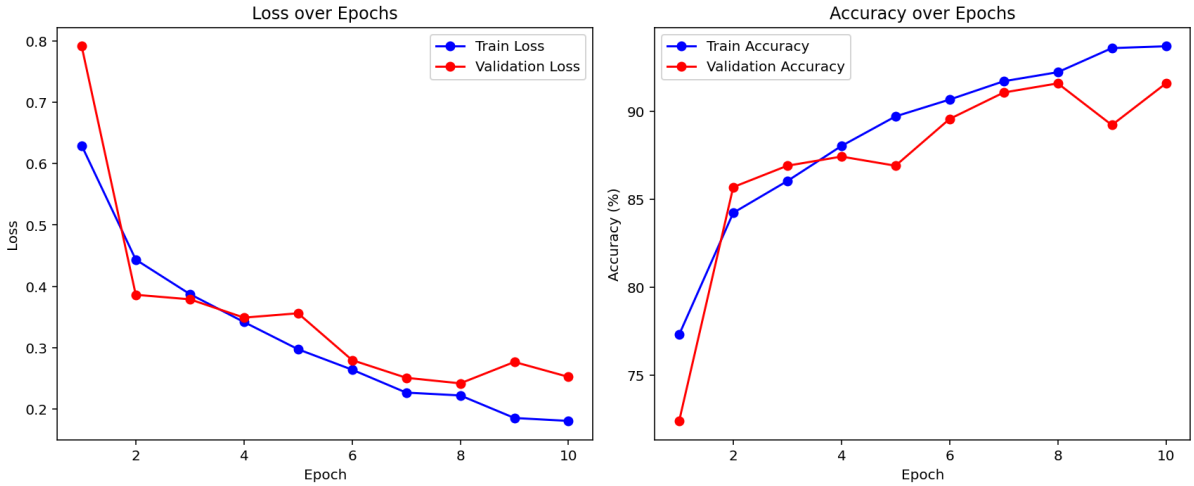


***Figure 13 – Epochs Impact on Lost & Accuracy***

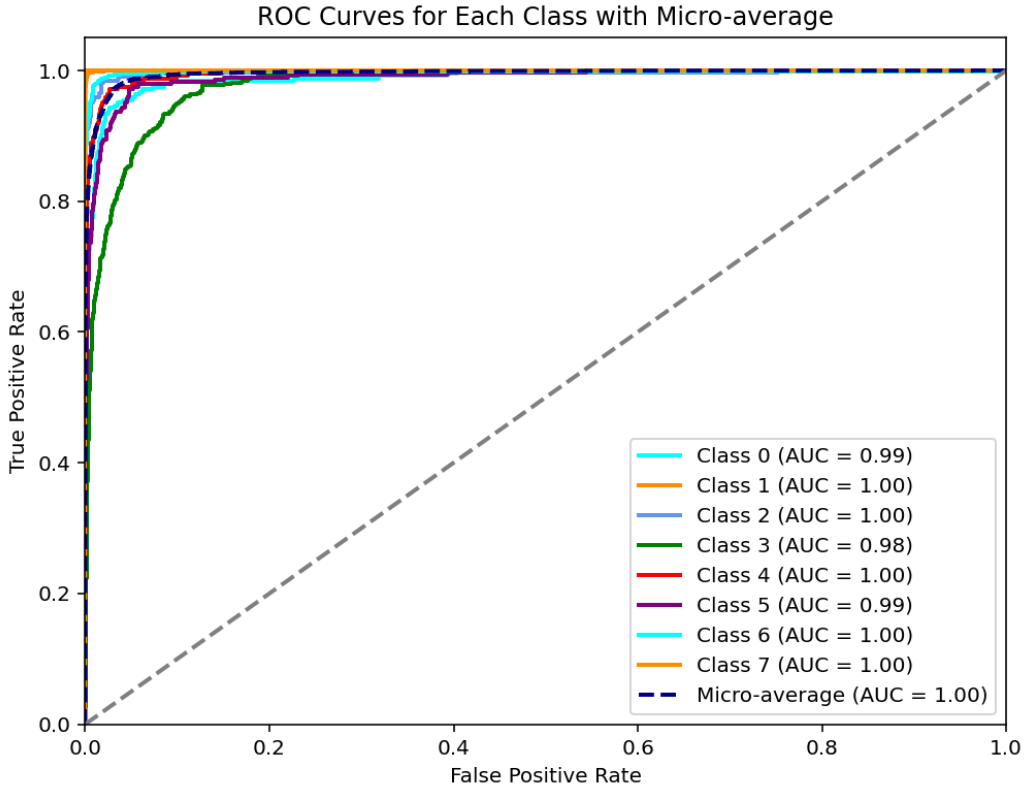


***Figure 14- ROC, BS=32, lr=0.005***

| class | Precision | Recall | F1-Score | Support |
|---|---|---|---|---|
| 0 | 0.913768 | 0.921053 | 0.717213 | 0.806452 |
| 1 | 0.913768 | 0.996737 | 0.979167 | 0.987874 |
| 2 | 0.913768 | 0.872464 | 0.967846 | 0.917683 |
| 3 | 0.913768 | 0.765321 | 0.884283 | 0.820513 |
| 4 | 0.913768 | 0.888889 | 0.888889 | 0.888889 |
| 5 | 0.913768 | 0.847518 | 0.841549 | 0.844523 |
| 6 | 0.913768 | 0.991776 | 0.905405 | 0.946625 |
| 7 | 0.913768 | 0.995754 | 0.997872 | 0.996812 |

***Table 4- ADAM Optimizer Result Table, lr = 0.005***

Now, we change the batch size to 32 with adam optimizer ewith learning rate of 0.001:

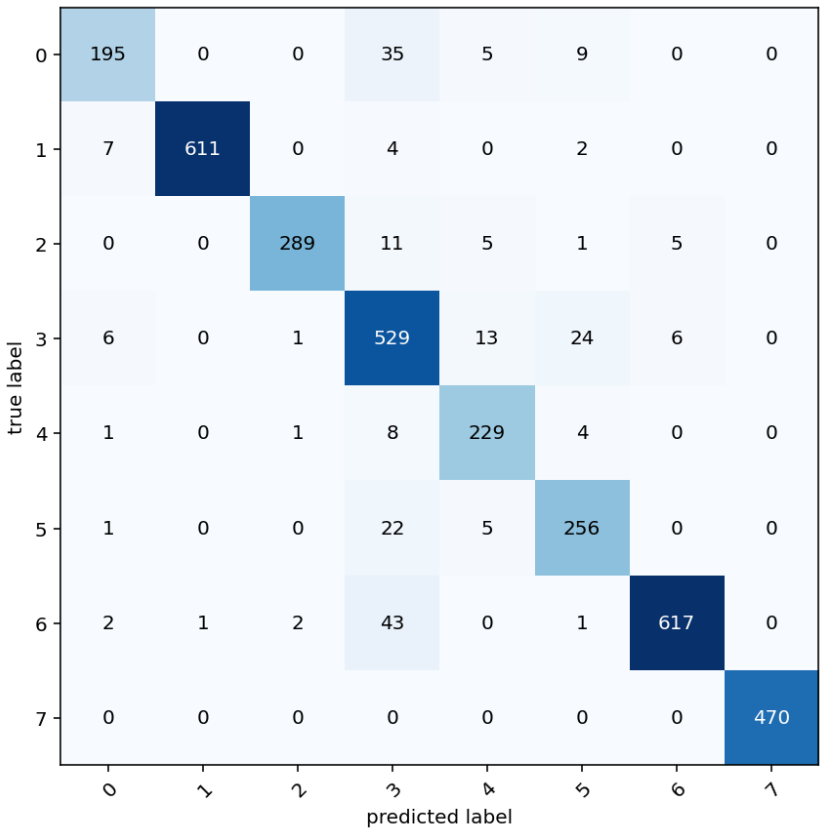


***Figure 15- Confusion Matrix, Adam Optimizer, BS=32, lr=0.001***

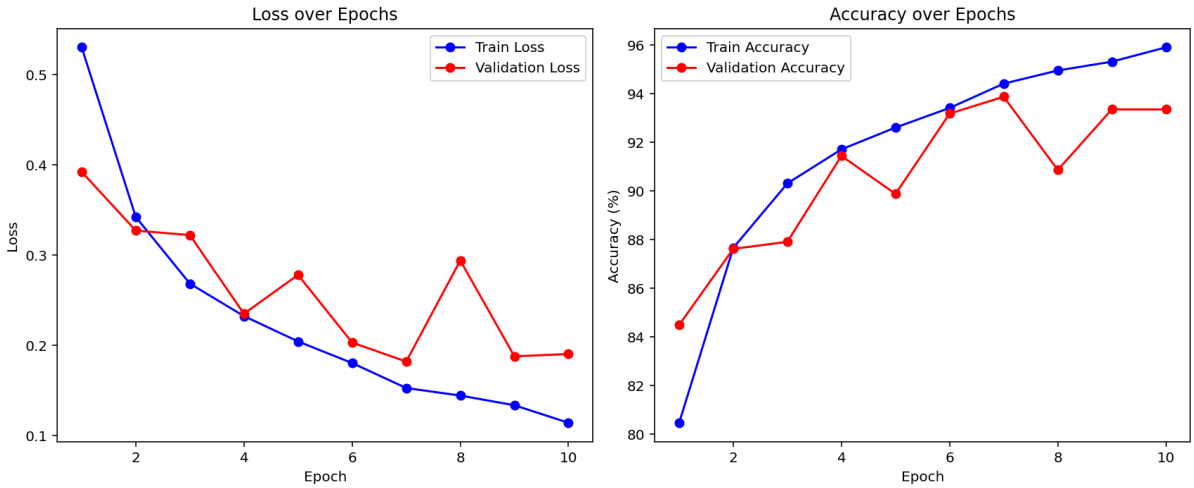


***Figure 16 – Epochs Impact on Lost & Accuracy***

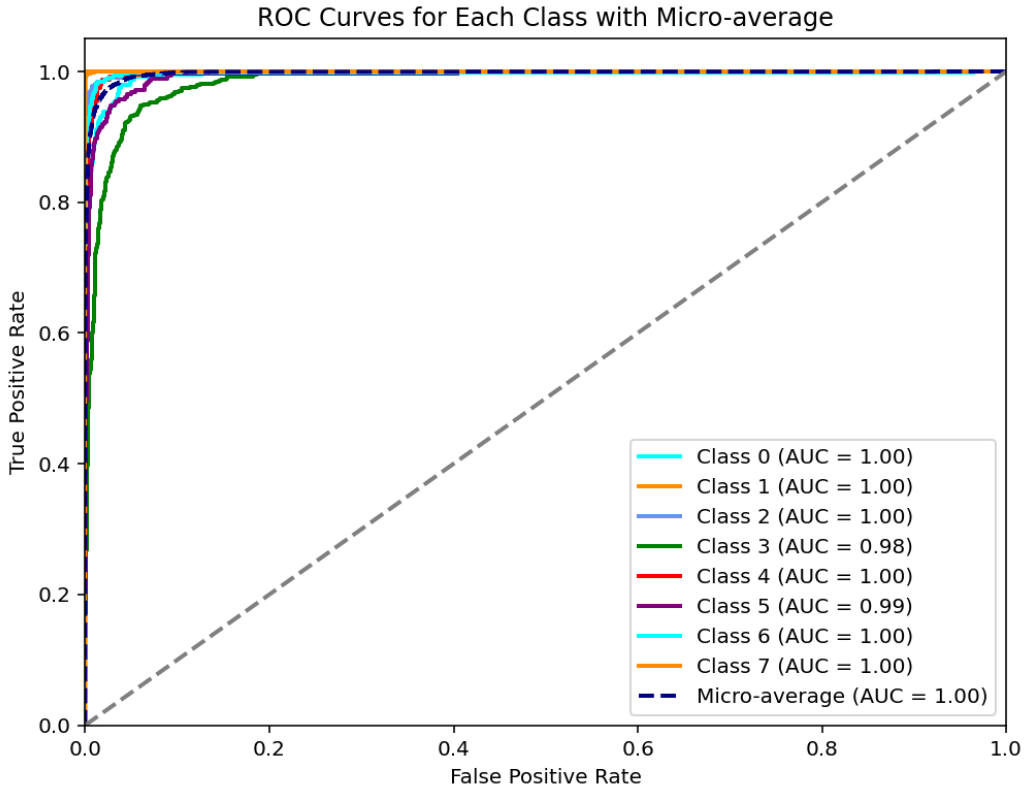


***Figure 17- ROC, BS=32, lr=0.001***

| class | Precision | Recall | F1-Score | Support |
|---|---|---|---|---|
| 0 | 0.93423 | 0.919811 | 0.799180 | 0.855263 |
| 1 | 0.93423 | 0.998366 | 0.979167 | 0.988673 |
| 2 | 0.93423 | 0.986348 | 0.929260 | 0.956954 |
| 3 | 0.93423 | 0.811350 | 0.913644 | 0.859464 |
| 4 | 0.93423 | 0.891051 | 0.942387 | 0.916000 |
| 5 | 0.93423 | 0.861953 | 0.9901408 | 0.881239 |
| 6 | 0.93423 | 0.982484 | 0.926426 | 0.953632 |
| 7 | 0.93423 | 1 | 1 | 1 |

***Table 5- ADAM Optimizer Result Table, lr = 0.001***

## 6. Discussion

The results indicate that while feature-based approaches such as Random Forest perform well, deep learning methods like CNNs offer significant improvements in accuracy and generalization. The CNN model demonstrated superior classification performance, particularly after hyperparameter tuning, suggesting that feature extraction directly from image data enhances accuracy. However, each method has its own trade-offs that must be considered.

One of the key strengths of the Random Forest classifier is its interpretability and robustness to small variations in data. Unlike deep learning models, Random Forest does not require extensive data preprocessing or large-scale training resources. Furthermore, the model provides insights into feature importance, allowing researchers to understand which aspects of the input data contribute most significantly to classification. Despite these advantages, the model's accuracy is limited by the expressiveness of manually engineered features, such as color histograms, which may fail to capture more complex spatial and morphological characteristics of blood cells.

In contrast, the CNN model exhibited superior performance due to its ability to automatically extract hierarchical features directly from raw image data. Convolutional layers capture spatial patterns, textures, and intricate structures, which are crucial for

distinguishing between similar cell types. The downside of CNNs, however, is their dependency on large amounts of labeled data and their computationally expensive training process. Additionally, hyperparameter tuning plays a critical role in optimizing CNN performance, requiring careful selection of network architecture, learning rates, batch sizes, and regularization techniques.

An important observation from the results is the class imbalance issue. In many medical imaging datasets, some classes may be underrepresented, leading to biased predictions**.** While CNNs demonstrated high accuracy overall, certain minority classes still faced misclassification. Addressing class imbalance through techniques such as data augmentation, SMOTE (Synthetic Minority Over-sampling Technique)**,** or **cost-**sensitive learning could further enhance model performance.

## 7. Conclusion

In conclusion, this study demonstrates that deep learning models outperform traditional machine learning approaches in blood cell classification tasks. The CNN model achieved significantly higher accuracy and AUC scores compared to the Random Forest classifier, highlighting the potential of deep learning for automated hematological diagnostics.

However, despite CNNs' superior performance, traditional machine learning models such as Random Forest still hold value in cases where computational resources are limited or where model interpretability is crucial. Future work should explore hybrid models that combine the interpretability of Random Forest with the feature learning capabilities of CNNs.

Moreover, efforts should be made to further improve CNN performance by incorporating transfer learning from larger medical imaging datasets. Using pre-trained models such as ResNet or EfficientNet could significantly enhance feature learning while reducing training time. Additionally, incorporating more advanced techniques such as attention mechanisms and explainability methods will be crucial for gaining trust in deep learning-based medical applications.

Overall, this study lays the groundwork for further exploration into automated blood cell classification, emphasizing the importance of both classical machine learning and modern deep learning techniques in advancing biomedical image analysis.

## 8. Future Work

While this study has demonstrated the effectiveness of machine learning and deep learning approaches in blood cell classification, several areas remain open for future research to enhance model performance, interpretability, and real-world applicability. Exploring more advanced deep learning architectures, such as ResNet, EfficientNet, or Vision Transformers (ViTs), could improve feature extraction and classification accuracy, while attention mechanisms and multi-scale feature learning may enable better focus on critical regions of blood cell images. Additionally, applying transfer learning from large-scale biomedical imaging datasets could help reduce training time and enhance model generalization, particularly when working with high-resolution blood cell images. Hybrid models that integrate traditional machine learning (e.g., Random Forest, SVM) with deep learning-based feature extraction could provide an optimal balance between interpretability and predictive power, allowing researchers to leverage both handcrafted features and learned representations. Addressing class imbalance remains a key challenge, which could be mitigated using oversampling techniques like SMOTE or ADASYN or by implementing cost-sensitive learning approaches to improve recall for minority classes. Furthermore, incorporating semi-supervised and self-supervised learning techniques could enable the use of unlabeled blood cell images, reducing dependence on manually labeled datasets. Another crucial aspect of future research is explainability and trust in AI-based diagnostics, where techniques such as Grad-CAM, SHAP, or LIME could be used to enhance the interpretability of CNN-based models, making them more accessible for medical professionals. Explainable AI (XAI) frameworks should be explored to bridge the gap between deep learning accuracy and the need for transparent, trustworthy decision-making in clinical practice. Lastly, deploying AI models into real-world medical applications by integrating them into automated hematology analyzers and testing them on diverse clinical datasets will be essential to ensure robustness and scalability in real-world diagnostics. By addressing these challenges, future research can contribute to the development of more accurate, interpretable, and efficient AI-driven hematology diagnostic tools, ultimately improving patient care and advancing the field of medical image analysis.

## 9. References


[1] Yang, J., Shi, R., Wei, D., et al. (2021). MedMNIST Classification Decathlon: A Lightweight AutoML Benchmark for Medical Image Analysis. IEEE 18th International Symposium on Biomedical Imaging (ISBI), 191–194.

[2] Tek, F.B., Dempster, A.G., & Kale, I. (2010). Malaria Parasite Detection Peripheral Blood Images. Computer Methods and Programs in Biomedicine, 101(2), 75–89.

[3] Breiman, L. (2001). Random Forests. Machine Learning, 45(1), 5–32.

[4] Pedregosa, F., Varoquaux, G., Gramfort, A., et al. (2011). Scikit-learn: Machine Learning in Python. Journal of Machine Learning Research, 12, 2825–2830.

[5] Esteva, A., Kuprel, B., Novoa, R.A., et al. (2017). Dermatologist-level classification of skin cancer with deep neural networks. Nature, 542(7639), 115–118.