White Blood Cell Sub-type detection

Krrish Jindal
Mathematics and Computing , IIT Roorkee

Machine learning has up and coming diverse applications in the healthcare industry especially when it comes to scans/images and computer vision. A particularly interesting area of research involves the classification of white blood cells. Accurate WBC classification is crucial for medical diagnosis but unfortunately right now hematologists have to manually classify WBCs by examining blood smears under a microscope, which is time-consuming and error-prone. To address this, we train several vision models for detection and diagnosis to reduce human error and speed up the classification process. We use an ensemble of various CNN models to classify WBCs. Our experimental analysis demonstrates that our approach achieves 98.7% accuracy in WBC detection and classification.
Keywords: Image classification, White blood cell detection, Deep learning, CNN, ensemble

1 Introduction

Human blood is composed of the blood cells which accounts for 45% of the blood tissue by volume, whereas the remaining 55% of the volume is composed of plasma (the liquid portion of the blood). There are three types of blood cells: red blood cells (erythrocytes), white blood cells (leukocytes), platelets (thrombocytes).
White blood cells are produced in the lymphoid tissue and bone marrow. These cells are classified into two types:
granulocytes and agranulocytes.
Granulocytes are further classified into neutrophils, basophils and eosinophils, whereas agranulocytes are classified into monocytes and lymphocytes, as shown in Fig. 1. Each type of cell fights against the infections or diseases. Neutrophils helps in healing the damaged tissues and resolve infections. Unusual levels of basophils result in severe types of blood disorders. Eosinophils are disease fighting white blood cells. Whereas, lymphocytes fight against bacteria, virus and the cells that threaten its functioning; monocytes fight against infections, remove dead or damaged tissues, destroy cancer cells. Safe to say that all these are really important thus being able to identify each of them becomes important for diagnostic purposes. WBCs have different shape of nucleus, color and texture, thereby making it a challenging task for manual analysis.

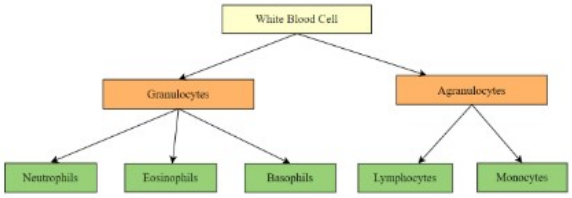

Figure 1: WBC classification

We classify into the following 4 classes.

- NEUTROPHIL
- EOSINOPHIL
- LYMPHOCYTE
- MONOCYTE

White blood cell counts are manually performed by hematologists. The process involves several steps to determine the location and classify the type of white blood cells. However, the current techniques for classifying white blood cells are prone to errors and time-consuming. To address this, computer-aided techniques are employed to automatically classify and locate white blood cells within blood cell images. Popular image classification models, such as CNN 1, 2, VGG16 3, and ResNet 4, play a crucial role in various applications and domains.

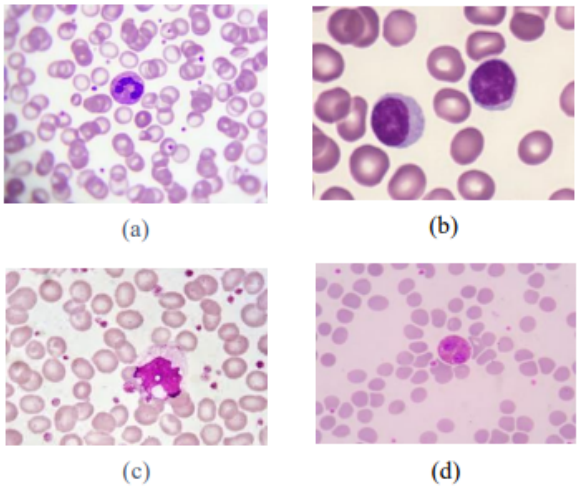

Figure 2: WBC classification: (a) Neutrophils (b) Lymphocytes (c) Monocytes (d) Eosinophils

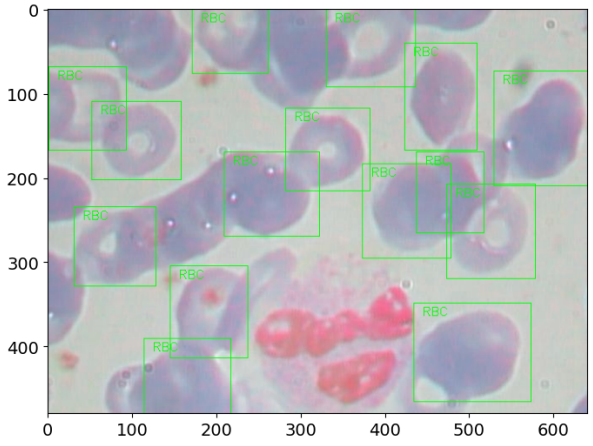

Figure 3: example

2 Related Work

There have been quite a few advances in the world of deep learning with respect to biology and even WBC detection and classification.

- Diouf et al. released a 7 layer CNN with standard convotutional and maxpooling layers to classify WBCs. [1] .
- Ozyurt proposed a CNN fused with MRMR feature selection. [2] .
- Vatathanavaro et al. explained the study of two CNN architectures: VGG-16 and ResNet-50 to classify five types of white blood cells. ResNet-50 is the best classifer and it achieved 88.3% accuracy. [3] .
- Theera-Umpon [13] utilized a new set of features generated using the mathematical morphology to extract nucleus features. Later, these features were used to train the artificial neural network and Bayes classifer to identify the type of WBC.[4]

3 Methodology

3.1 Data Collection, Preprocessing, and Augmentation

The study utilizes a dataset comprising images of blood cells sourced from a publicly available repository. Each image undergoes meticulous annotation to distinguish the cell-of-interest, marked in purple, from other blood cells colored in a distinct skin-red shade. This annotation ensures the focus remains on a specific cell type, facilitating accurate classification.
Following acquisition, the images undergo rigorous preprocessing to standardize their format and reduce potential biases. Resizing the images to a uniform dimension of 224×224 pixels ensures computational efficiency while retaining crucial details. Additionally, pixel normalization addresses variations in lighting conditions across different images, promoting consistency in feature representation.
Data augmentation techniques are employed to enhance dataset diversity and improve model robustness. Random transformations such as rotation, flipping, and zooming are applied to simulate real-world variations and mitigate overfitting. Augmenting the dataset exposes the models to a broader range of scenarios, enabling better generalization to unseen data.

3.2 Model Architectures and Training

3.2.1 Custom Convolutional Neural Network (CNN)

In this study, we train our custom CNN. Our proposed CNN architecture consists of a sequential stack of convolutional, pooling, and dense layers, interspersed with batch normalization and dropout regularization. The rationale behind:
Architecture Details

- **Input Layer:** The network starts with an input layer that accepts images of size 224×224 with 3 color channels.
- **Convolutional Layers:** Multiple convolutional layers are employed to extract spatial features at different scales. Each convolutional layer uses ReLU activation functions to introduce non-linearity and padding strategies to preserve spatial dimensions.
- **Batch Normalization:** After each convolutional layer, batch normalization is applied to standardize and stabilize the activations, aiding in faster convergence during training.

- **Max Pooling Layers:** Max pooling layers are inserted to downsample feature maps, facilitating translation invariance and reducing computational complexity.
- **Dense Layers:** The flattened feature maps go through fully connected dense layers that serve as the final stages of feature aggregation and abstraction.
- **Dropout Regularization:** Dropout regularization is used to mitigate overfitting by deactivating neurons during training.
- **Output Layer:** The output layer is a dense layer with softmax activation, yielding probabilistic predictions across the target classes.

Training Procedure

- **Optimizer:** Stochastic gradient descent (SGD) is used as the optimizer to minimize the categorical cross-entropy loss function.
- **Learning Rate Scheduler:** A learning rate scheduler dynamically adjusts the learning rate during training, allowing for finer control over the optimization process.
- **Batch Training:** The training data is passed in batches, with each batch having a fixed number of images.
- **Epochs:** The training process iterates over the entire dataset ten times, with model parameters updated after each epoch to minimize the loss function.
- **Validation:** Model performance is evaluated on a validation set during training to monitor for overfitting and ensure generalization.
- **Total Prarmeters:**he model has a total of 15,611,524 parameters, out of which 15,605,124 are trainable and 6400 are non-trainable.

3.2.2 MobileNet Model

MobileNet is a lightweight convolutional neural network architecture designed for mobile and embedded vision applications. It consists of depthwise separable convolutions, which significantly reduce the computational cost and model size while maintaining high accuracy.

Architecture Details

- **Pre-Trained Model::** MobileNetV2 is pre-trained on the ImageNet dataset, leveraging the knowledge learned from a large-scale image classification task.
- **Fine-Tuning:** The pre-trained MobileNetV2 model is fine-tuned on the blood cell dataset by adding additional dense layers for classification.
- **Batch Normalization:** After each convolutional layer, batch normalization is applied to standardize and stabilize the activations, aiding in faster convergence during training.
- **Global Average Pooling:** Global average pooling is used to aggregate spatial information from the feature maps, reducing the dimensionality of the data before the final classification layer.

Training:

- **Optimizer:** Adam optimizer with a learning rate of 0.001 is used to minimize the categorical cross-entropy loss function during training.
- **Fine-Tuning:** The pre-trained MobileNetV2 model is fine-tuned on the blood cell dataset by adding additional dense layers for classification.
- **Epochs:** The model is trained over 10 epochs, with early stopping implemented to prevent overfitting.

3.2.3 ResNet Model

ResNet50 is a deep convolutional neural network architecture renowned for its effectiveness in image classification tasks. It introduces residual connections, it allows for the training of deeper networks without suffering from vanishing gradient problems.

Architecture Details: We introduce a customized CNN architecture rooted in the ResNet50 framework. Our design revolves around residual blocks, featuring both shortcut and main paths. Within each residual block, the main path incorporates convolutional layers complemented by batch normalization and ReLU activation, while the shortcut path employs 1x1 convolutions followed by batch normalization. By stacking these residual blocks, we form the network, adjusting the number of filters and strides as needed for each block. This configuration allows for deeper and more efficient feature extraction.

Training: Following architecture definition, our model undergoes standard training procedures. We initialize model parameters and optimize them using a specified loss function, in our case categorical cross-entropy. We employ the Adam optimizer with a learning rate of 0.001 for gradient descent. During training, we utilize a dataset suited to our task, partitioning it into training and validation sets for model evaluation. The training process involves iterating over the dataset for a predefined number of epochs, updating model weights to minimize the loss function. Through this approach, our

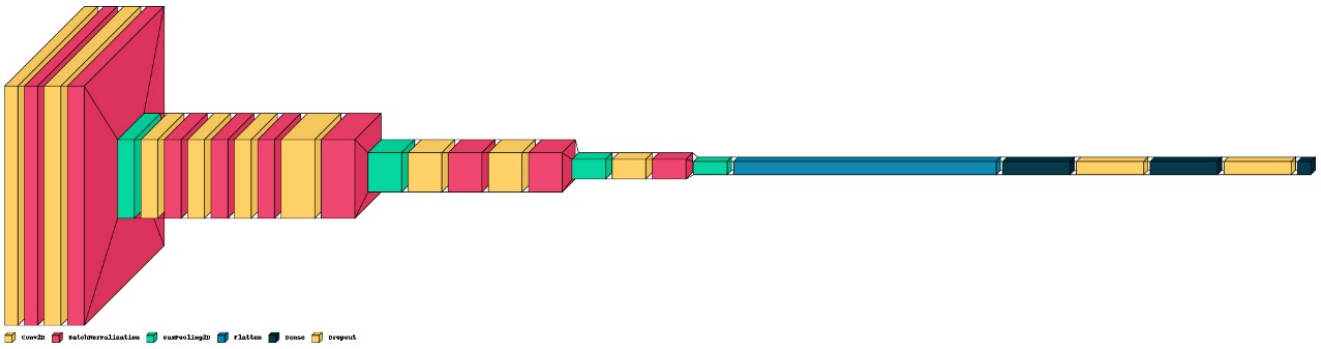

Figure 4: CNN that we made

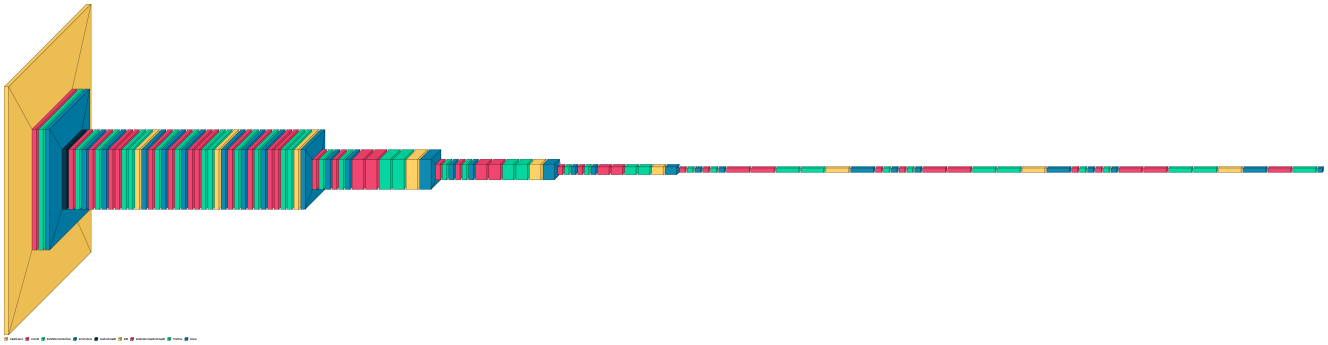

Figure 5: Resnet

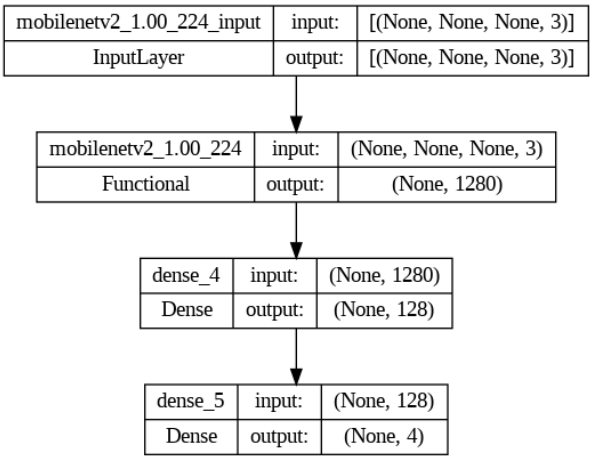

Figure 6: mobileNet Architecture

model learns to accurately classify input data, demonstrating competitive performance on the given task.

3.3 Model Ensemble

In order to improve the accuracy of classification, a combination of the two best performing models , the custom CNN and the MobileNet was used through an ensemble approach. The blending of predictions in the ensemble model was achieved by combining the outputs of the custom CNN and MobileNet models using a simple voting mechanism. This method involved summing up the predictions from both models and selecting the class with the highest total prediction score as the final prediction. This allowed the ensemble model to make use of diverse strengths of each individual model, resulting in higher accuracy and robustness.
.

3.4 Effect of optimizers

The success of deep learning models heavily relies on the optimization process, which involves updating the model parameters iteratively to minimize the loss function. SGD, the classical optimization algorithm, and Adam, a more recent variant, are two widely used optimizers in the field of deep learning. SGD updates the parameters based on the gradient of the loss function computed on a mini-batch of training data, while Adam incorporates adaptive learning rates and momentum to accelerate convergence. Understanding the characteristics and performance of these optimizers is essential for practitioners and researchers to make informed decisions regarding model training.
Key Characteristics of SGD:

- SGD computes the gradient of the loss function with respect to model parameters using a mini-batch of training data.
- It updates the parameters in the opposite direction of the gradient multiplied by the learning rate.

$$w_{t+1} = w_t - \eta \frac{\partial J(w)}{\partial w_t} \tag{1}$$

- SGD does not incorporate adaptive learning rates

or momentum, making it simple and easy to implement.

- SGD may suffer from slow convergence and oscillations, especially in the presence of noisy or ill-conditioned gradients.

Key Characteristics of Adam Optimizer:

- Adam combines the advantages of adaptive learning rates and momentum to improve convergence speed and stability.

$$w_{t+1} = w_t - \alpha * m_t \tag{2}$$

where,

$$m_t = \beta m_{t-1} + (1 - \beta) * \frac{\partial L}{\partial w_t} \tag{3}$$

- It maintains separate learning rates for each parameter and adapts them based on the first and second moments of the gradients.
- Adam incorporates bias correction to compensate for the initialization bias and reduce the impact of the first few iterations.
- Adam is well-suited for a wide range of deep learning tasks and often achieves faster convergence compared to traditional optimization algorithms.

Conclusion: In conclusion, both Stochastic Gradient Descent and Adam optimizer are essential tools in the deep learning toolbox, each with its unique strengths and weaknesses. While SGD offers simplicity and efficiency, Adam provides adaptive learning rates and momentum to accelerate convergence and improve performance. Understanding the trade-offs between these optimizers and selecting the appropriate one for a given task is crucial for achieving optimal results in deep learning applications.

4 Performance Evaluation

The performance of the WBCs classification is evaluated using accuracy on the test set and a confusion matrix. Table 1 contains different performance metrics of the individual models and the final ensemble.

4.1 CNN performance

After evaluating the test dataset, we found that the first CNN model achieved an accuracy of 96.2% with a categorical cross-entropy loss of 0.1078. The classification report indicated high precision, recall, and F1-scores for all classes. Additionally, we have provided graphs that show the training and validation set accuracy and loss curves.

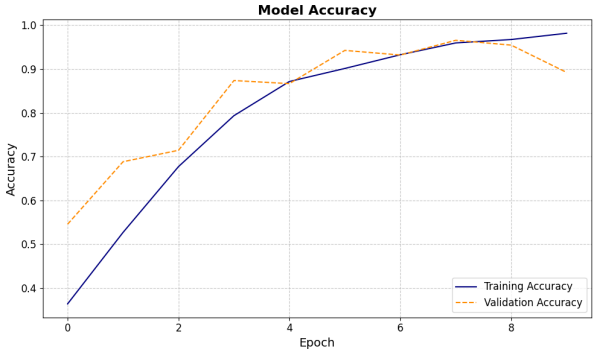

Figure 7: CNN training accuracy

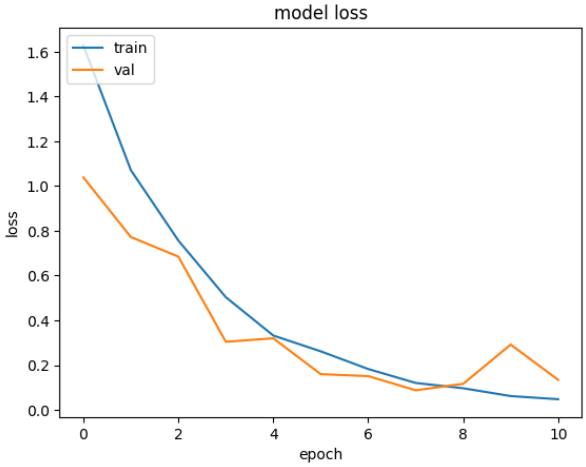

Figure 8: CNN training loss

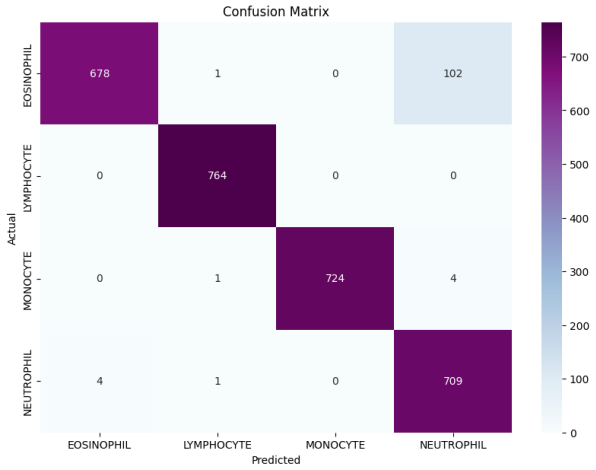

Figure 9: CNN confusion matrix

Table 1: Classification report of CNN

	Precision	Recall	F1-score	Support
EOSINOPHIL	0.99	0.87	0.93	781
LYMPHOCYTE	1.00	1.00	1.00	764
MONOCYTE	1.00	0.99	1.00	729
NEUTROPHIL	0.87	0.99	0.93	714

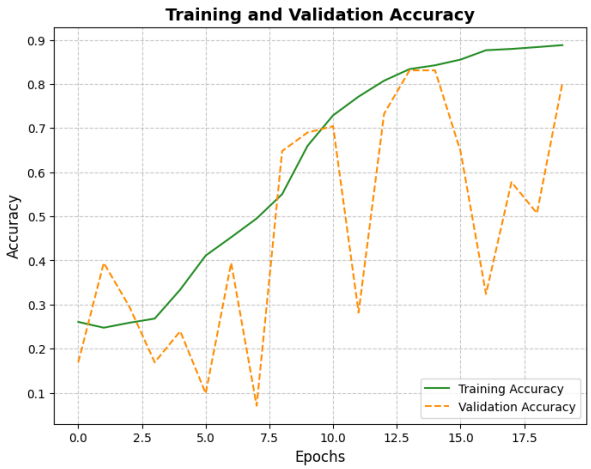

Figure 10: ResNet training accuracy

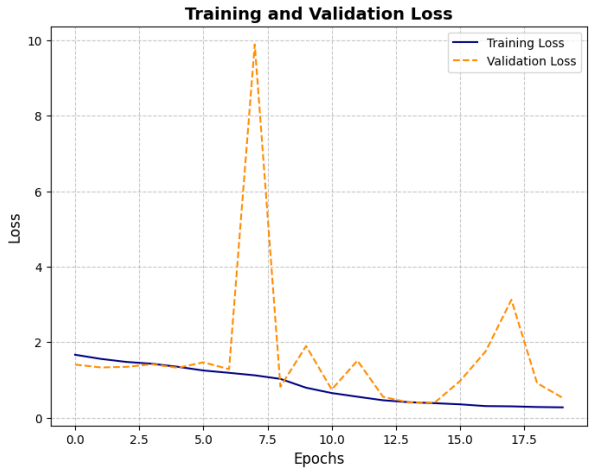

Figure 11: ResNet training loss

4.2 ResNet performance

The evaluation of the ResNet model demonstrated an accuracy of 86%. The classification report indicated variations in precision, recall, and F1-score across different classes. Although ResNet maintained high precision and recall for the lymphocyte class, it exhibited relatively lower performance for the neutrophil class.We have provided graphs that show the training and validation set accuracy and loss curves.

4.3 MobileNet Performance

The MobileNet model yielded an accuracy of 93.% on the test set, accompanied by a loss of 0.1713. Although

Table 2: Summary of model accuracies in blood cell image classification.

Model	Accuracy
CNN	96.2%
ResNet	86%
MobileNet	93.8%
Ensemble	98.7%

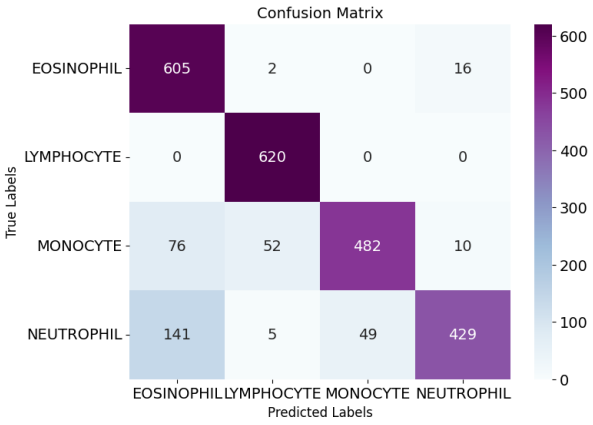

Figure 12: ResNet confusion matrix

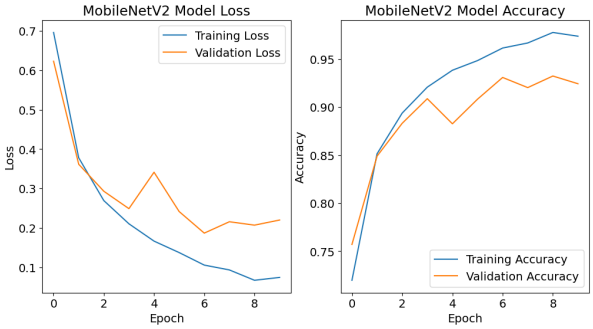

Figure 13: MobileNet training loss and accuracy

slightly lower in accuracy compared to the 1st CNN, MobileNet demonstrated commendable performance across all classes, as evidenced by the classification report.we have provided graphs that show the training and validation set accuracy and loss curves.

4.4 Ensemble Performance

Combining the predictions of the individual models through ensembling resulted in a significant improvement in accuracy, achieving 98.7The ensemble model exhibited superior performance across all classes, with high precision, recall, and F1-scores, highlighting the effectiveness of ensemble learning in enhancing classification accuracy.

5 Conclusions

We conducted a thorough investigation into the classification of blood cell images using convolutional neural networks (CNNs) and ensemble learning techniques. We analyzed various models, including a custom CNN architecture, MobileNet, and ResNet.
Our custom CNN architecture outperformed other models, achieving an impressive 96.2% accuracy on the test dataset. Ensemble learning further increased the classification accuracy to 98.7%, demonstrating the effectiveness of combining diverse model predictions.

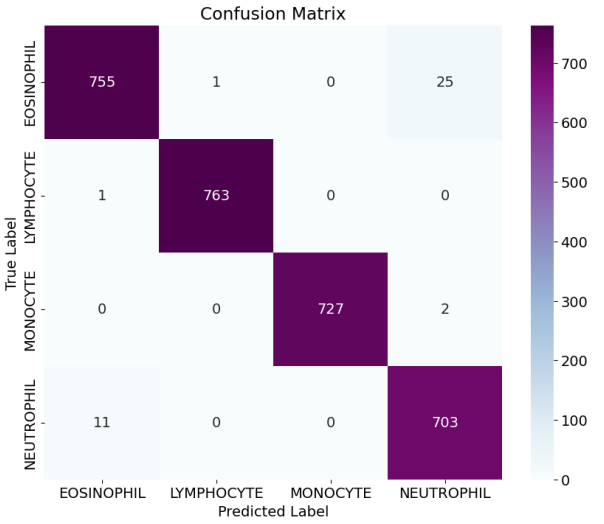

Figure 14: Ensemble confusion matrix

We also examined hyperparameters such as batch normalization and found that it plays a crucial role in improving model convergence and generalization.
Our study highlights the importance of fine-tuning hyperparameters and selecting appropriate architectures for achieving optimal performance in medical image classification tasks. The demonstrated accuracy and reliability of our approaches suggest their potential for aiding medical diagnostics and research.
Overall, our study provides valuable insights into the application of deep learning methodologies for blood cell classification, with implications for medical diagnostics and research. The high accuracy and reliability of our proposed approaches suggest their potential to enhance healthcare systems by facilitating efficient and accurate analysis of blood cell images, which can aid in the diagnosis and treatment of various hematologic disorders.
Looking forward, continued research and development in this field hold promise for further advancements in automated medical image analysis and healthcare delivery.

6 Declarations

6.1 Acknowledgements

We acknowledge IIT Roorkee and the Government of India for providing the Param Ganga cluster where we conducted some of our experiments.

References

[1] D. Diouf, M. Diop, and A. Ba, *Convolutional neural network and decision support in medical imaging: case study of the recognition of blood cell subtypes*.

[2] F. Ozyurt, *A fused cnn model for wbc detection with mrmr feature ¨ selection and extreme learning machine*.

[3] S. Vatathanavaro, S. Tungjitnob, , and K. Pasupa, *White blood cell classification: A comparison between vgg-16 and resnet-50 models*.

[4] N. Theera-Umpon, *Automatic white blood cell classification in bone marrow images using morphological features*.