HematoVision

Blood Cell Classification Using Transfer Learning

By

Kapuluri Varshini (S200807, RGUKT-Srikakulam)

Hemanth Yarra (S200406, RGUKT-Srikakulam)

G. Subhan Basha (22091A04K7, RGMCET-Nandyal)

Date

June 25, 2025

Abstract

Automated blood cell classification plays a vital role in medical diagnostics by supporting pathologists in identifying different types of white blood cells. In this project, we propose a deep learning-based approach using transfer learning to classify four types of white blood cells—eosinophils, lymphocytes, monocytes, and neutrophils—from microscopic images. We employed MobileNetV2, a lightweight and efficient convolutional neural network pretrained on ImageNet, as the backbone model. By customizing the classifier head and applying dropout and regularization, the model was optimized for multiclass classification while minimizing overfitting.

The dataset used in this project is sourced from Kaggle and consists of labeled images of white blood cells. Instead of using the provided validation set, we combined it with the training set and internally split it to better control the training and validation process. Data augmentation techniques such as rotation and zoom were applied to increase diversity and improve generalization. The final model achieved high classification accuracy and demonstrated strong performance on unseen test data, suggesting its potential utility in medical image analysis tasks.

Contents

1 Introduction

Blood cell classification plays a critical role in clinical diagnostics, especially for detecting hematological and immunological conditions. Traditionally, this process involves manual examination of microscopic images by hematologists, which is labor-intensive, time-consuming, and prone to human error. Recent advancements in deep learning and computer vision offer promising solutions to automate this task with high accuracy and consistency. In this context, HematoVision is a deep learning-based system designed to classify white blood cells efficiently using a lightweight architecture. The use of transfer learning with MobileNetV2 provides an optimal balance between speed and performance, making the system scalable and suitable for real-world deployment.

2 Objective

The primary objective of this project is to develop a reliable and efficient image classification system to automatically identify four types of white blood cells—eosinophils, lymphocytes, monocytes, and neutrophils—from microscopic images. Key goals include:

- Achieving high classification accuracy using transfer learning.
- Building a user-friendly web application for image upload and prediction.
- Ensuring fast, real-time inference for practical use in diagnostics and training.

3 Dataset

The dataset used in this project is sourced from Kaggle's Blood Cell Images repository, which provides a diverse collection of microscopic images of white blood cells. It is divided into two main folders: TRAIN and TEST. The training set contains approximately 12,500 images, while the test set includes around 3,000 images, all organized into four major classes:

- **Eosinophils**: Bi-lobed nucleus, granules stained reddish-orange.
- **Lymphocytes**: Dense, large nucleus with thin cytoplasm rim.
- **Monocytes**: Kidney-shaped nucleus, phagocytic behavior.
- **Neutrophils**: Multi-lobed nucleus, pale pink cytoplasmic granules.

Each image is labeled, and the dataset is well-suited for supervised classification tasks. The Training folder was internally split into training and validation subsets using ImageDataGenerator with a validation split. The Testing folder was used exclusively for evaluating the final model's performance.

This dataset offers good inter-class variability and balanced class distribution, making it suitable for training deep learning models effectively in the domain of hematological image classification

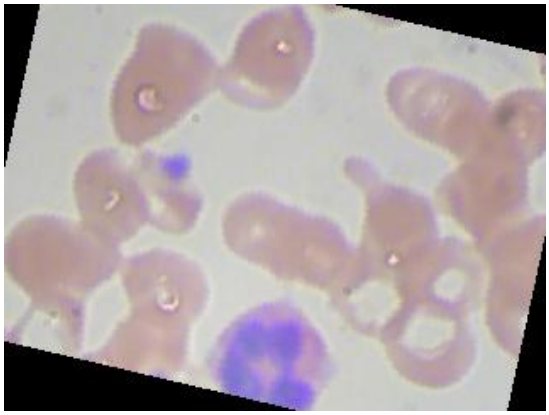

Figure 1: Eosinophil

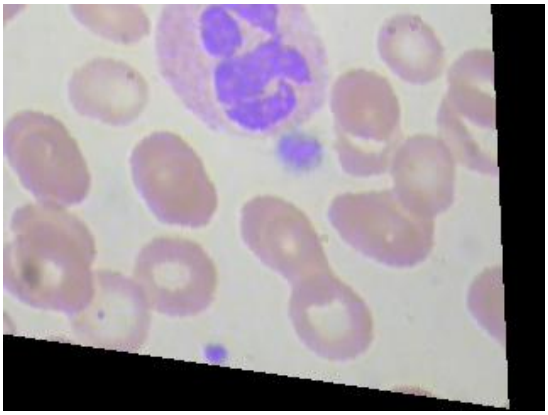

Figure 2: Lymphocyte

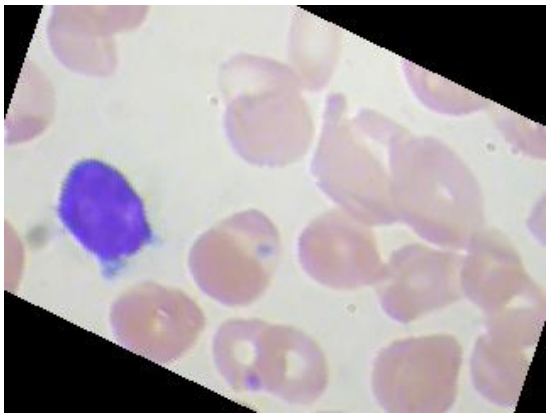

Figure 3: Monocyte

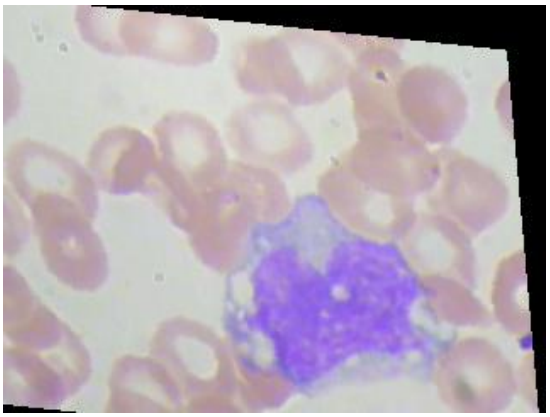

Figure 4: Neutrophil

4 Methodology – Preprocessing

Before training the deep learning model, the dataset underwent essential preprocessing to ensure consistency and enhance model performance. The following steps were performed:

Resizing: All images were resized to **128×128 pixels** to maintain uniform input dimensions compatible with the MobileNetV2 architecture.

Normalization: Pixel values were rescaled to the range **[0, 1]** by dividing by 255, improving convergence during training.

Data Augmentation: To increase dataset diversity and reduce overfitting, various augmentation techniques were applied to training images, including:

- Random rotation (up to 30 degrees)
- Zooming (up to 30%)
- Width and height shifts (up to 20%)

- Shearing
- Horizontal flipping
- Filling missing pixels using the nearest mode

These transformations were implemented using **Keras'** `ImageDataGenerator`. The training dataset was split into **80% training** and **20% validation** internally. The test set was used as-is without augmentation for final evaluation.

5 Model Architecture

To achieve efficient and accurate blood cell classification, **MobileNetV2** was adopted as the backbone using a transfer learning approach. MobileNetV2 is a lightweight convolutional neural network pre-trained on ImageNet, known for its balance between performance and computational efficiency—ideal for medical image tasks.

Training Pipeline Overview

Base Model: MobileNetV2 loaded without the top classification layer (`include_top=False`) and pre-trained on ImageNet.

Input Shape: All input images were resized to `128 × 128 × 3`.

Custom Head:

- `GlobalAveragePooling2D` layer to reduce spatial dimensions.
- `Dropout` layer with a rate of 0.5 to prevent overfitting.
- `Dense` output layer with 4 units and `Softmax` activation for multi-class classification.

Loss Function: `categorical_crossentropy`, suited for one-hot encoded labels.

Optimizer: `Adam` optimizer with a learning rate of `1e-4` during base training, reduced to `1e-5` during fine-tuning.

Training Strategy:

- **Phase 1:** Freeze base model and train only custom layers for 10 epochs.
- **Phase 2:** Unfreeze and fine-tune the entire model for 10 additional epochs.

Callbacks Used:

- `ModelCheckpoint` – saves the best model during training.
- `EarlyStopping` – stops training if validation loss doesn't improve.
- `ReduceLROnPlateau` – lowers the learning rate on validation loss plateau.

This architecture ensures both generalization from transfer learning and adaptation to the specific blood cell dataset.

6 Results and Evaluation

The MobileNetV2-based model demonstrated strong performance in classifying white blood cell images. After training for 20 epochs using transfer learning and fine-tuning, the model achieved a test accuracy of **85.52%** and a test loss of **0.4704**.

The training and validation accuracy were plotted over epochs to monitor model learning dynamics. Figure **??** shows how the validation accuracy closely follows the training accuracy, suggesting the model generalizes well without overfitting.

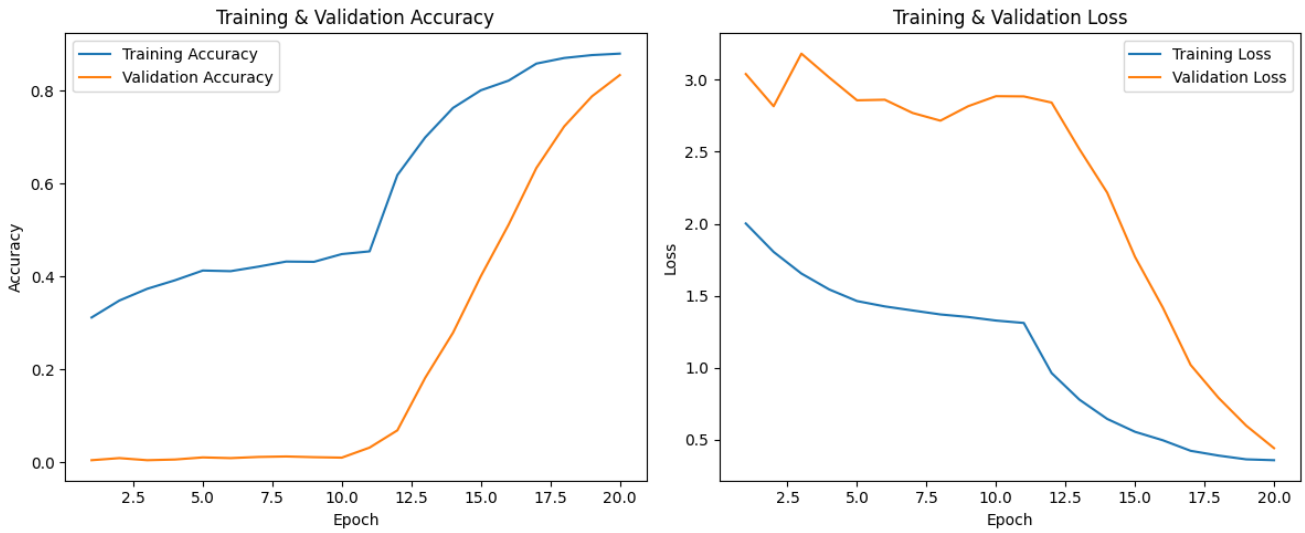

Figure 5: Training vs Validation Accuracy and Loss Curves

This visualization reinforces that the model steadily improved and converged during both training and fine-tuning phases.

To better understand model performance, a classification report was generated and is summarized in Table 1.

Table 1: Classification Report

Class	**Precision**	**Recall**	**F1-score**
Eosinophil	0.90	0.69	0.78
Lymphocyte	0.86	0.99	0.92
Monocyte	0.92	0.91	0.92
Neutrophil	0.85	0.86	0.85

The F1-scores indicate the model's robustness across all classes, particularly excelling in monocyte and lymphocyte classification.

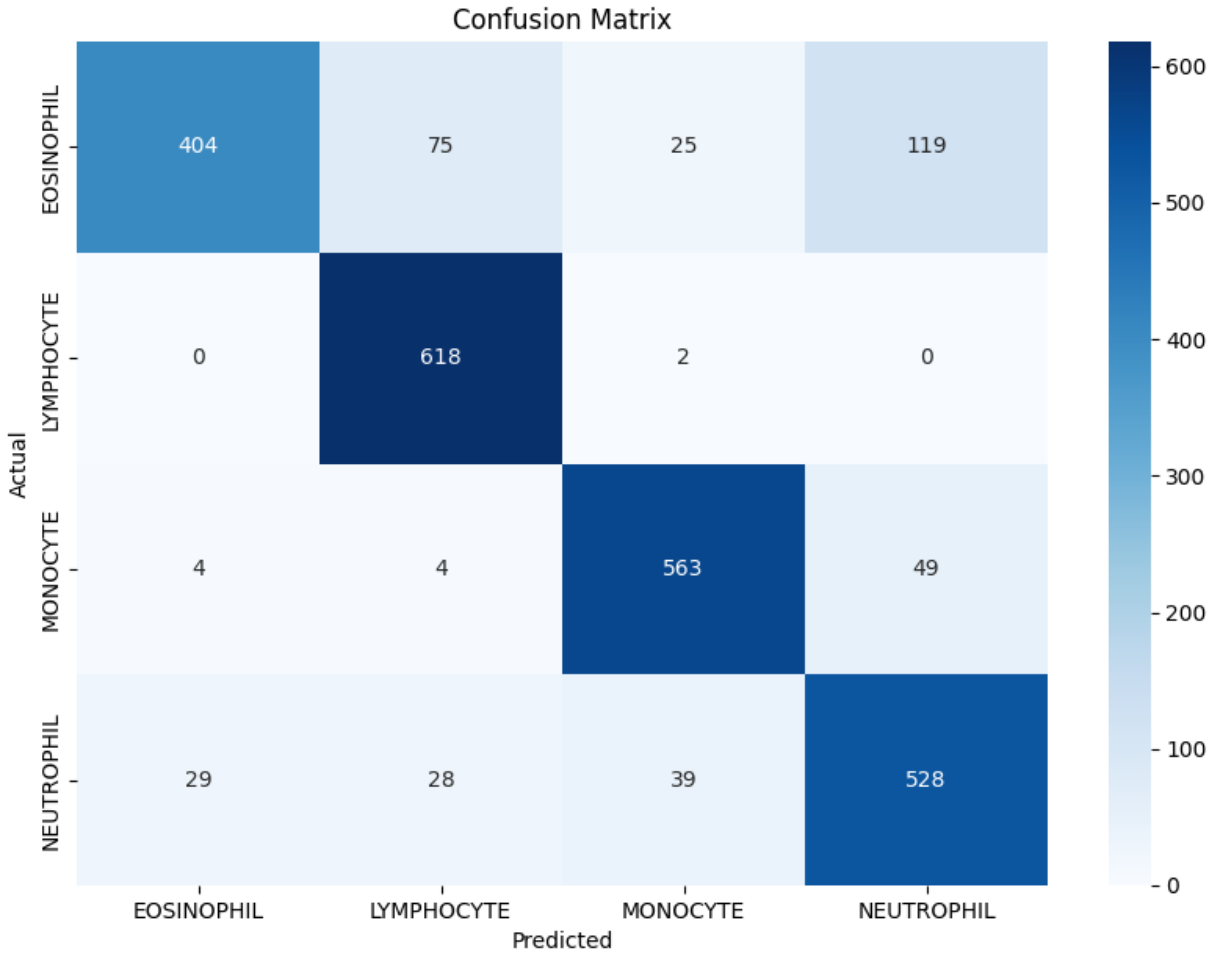

Figure 6: Confusion matrix

7 Applications

HematoVision offers valuable use across multiple domains:

- **Clinical Diagnostics:** Enables faster, automated pre-screening of blood samples, reducing manual workload.
- **Telemedicine:** Supports remote diagnosis by processing uploaded microscope images via mobile or web platforms.
- **Medical Education:** Acts as a training tool with real-time feedback, helping students learn blood cell classification effectively.

8 Flask Deployment

The trained model is deployed using a lightweight Flask app.

System Flow

1. User uploads an image via the web UI.
2. The image is saved and processed with OpenCV.
3. Model prediction is generated using Keras.
4. The result is displayed with the image using base64 encoding.

Code Overview

The backend includes:

- `app.py`: Flask logic for file handling and prediction
- `home.html` & `result.html`: Frontend templates
- `blood_cell.h5`: Trained CNN model

Image preprocessing uses OpenCV and MobileNetV2-specific normalization. The final label is inferred using `np.argmax()` and mapped to a class.

Working Demo

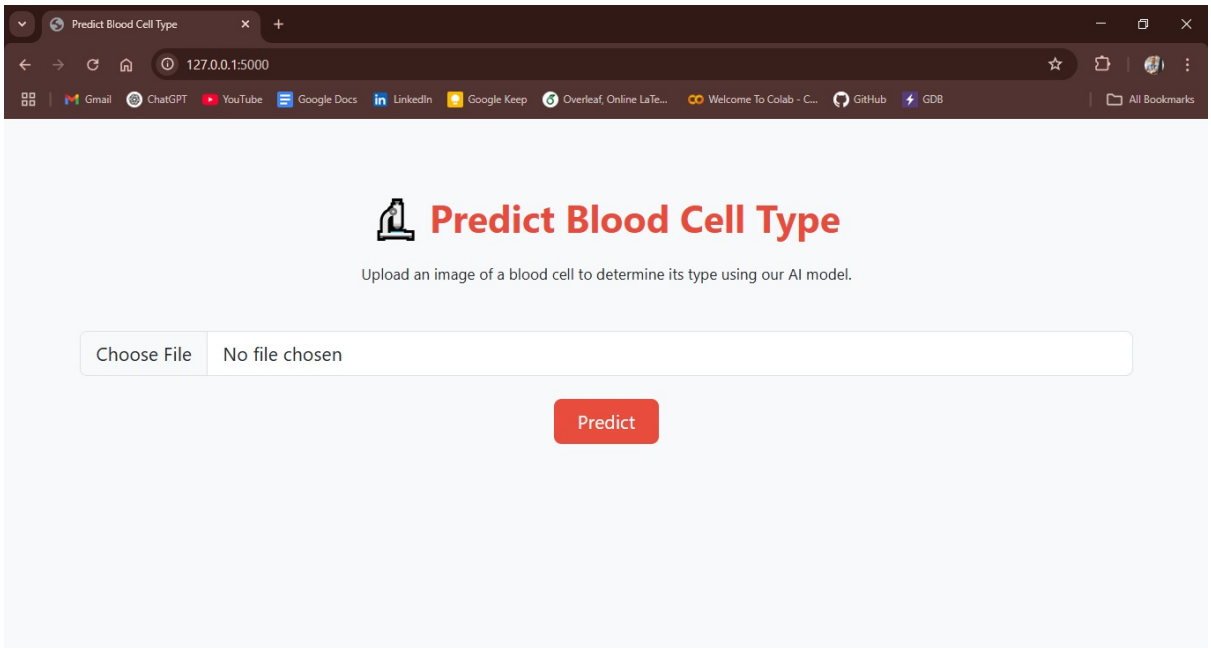

Figure 7: Home page of the Website

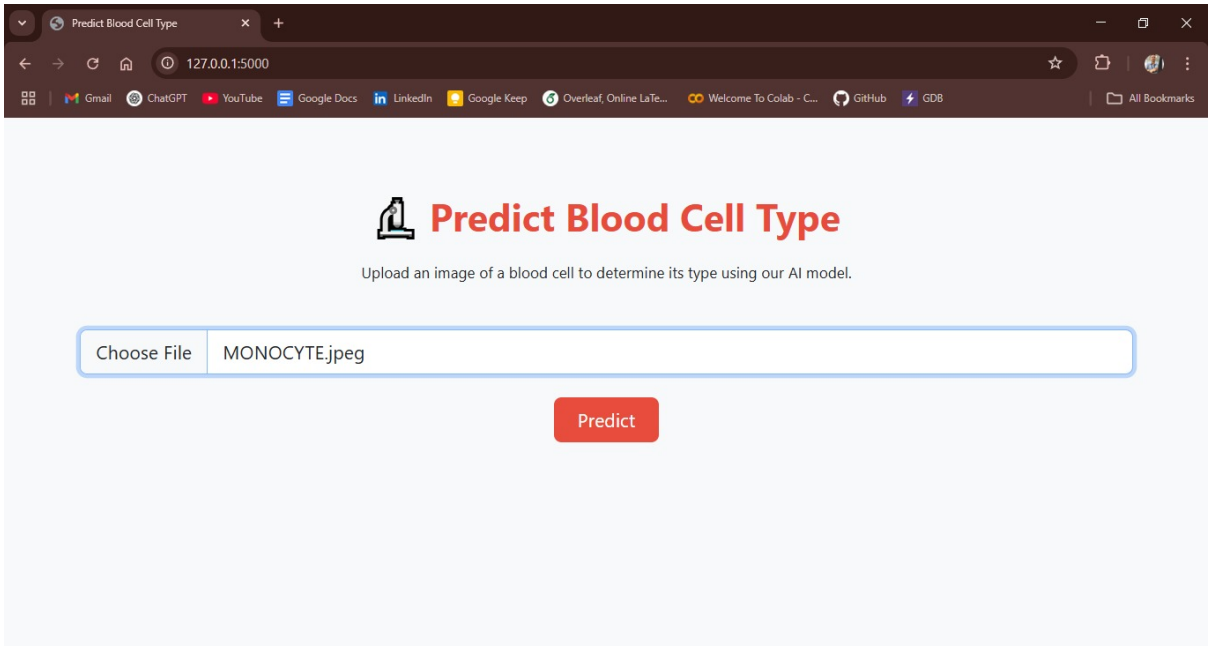

Figure 8: Image Uploading

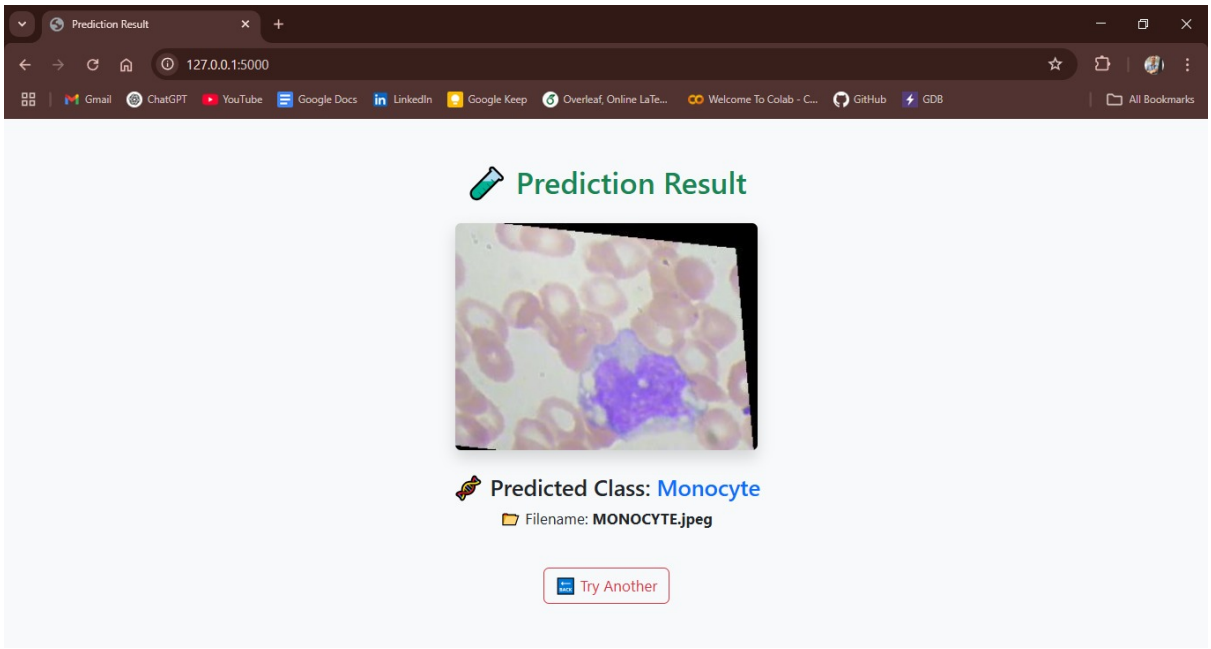

Figure 9: Predicting Result

9 Conclusion

HematoVision effectively applies transfer learning to the task of blood cell classification, achieving high accuracy and efficiency using MobileNetV2. The model's lightweight architecture and performance make it suitable for both clinical diagnostics and educational purposes. With its successful deployment in a user-friendly interface, HematoVision proves to be a practical tool for real-time medical image analysis.

10 Future Work

To further enhance HematoVision, future efforts may include:

- Expanding the model to classify rarer or more nuanced blood cell subtypes.
- Deploying the system as a cloud-based API using platforms like Streamlit or Google Cloud Platform.
- Integrating user authentication and data security features for clinical-grade deployment.

References

[1] S. J. Pan and Q. Yang, "A Survey on Transfer Learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, no. 10, pp. 1345–1359, 2009.

[2] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L. Chen, "MobileNetV2: Inverted Residuals and Linear Bottlenecks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[3] A. Rehman, S. Naz, and A. Razzak, "Classification of white blood cells using deep features and machine learning," *Journal of Medical Systems*, vol. 44, 2020.

[4] P. Mooney, "Blood Cell Images Dataset," Kaggle, 2018. [Online]. Available: `https://www.kaggle.com/datasets/paultimothymooney/blood-cells`

[5] HematoVision GitHub Repository, [Online]. Available: `https://github.com/Yarra-Hemanth/HematoVision`

[6] HematoVision Live Demo, [Online]. Available: `https://hemanth-bloodcells.streamlit.app`