HematoVision:

Advanced Blood Cell Classification Using Transfer Learning

1. Introduction

1.1 Project Overview

This project aims to develop a deep learning model that classifies blood cells into four major types: **eosinophils, lymphocytes, monocytes**, and **neutrophils**. Leveraging **transfer learning** using the **MobileNetV2** model and a dataset of annotated blood cell images, the goal is to assist healthcare professionals with faster and more accurate diagnostics.

1.2 Objectives

- Collect and preprocess a dataset of labeled blood cell images.
- Apply transfer learning using MobileNetV2 for image classification.
- Train, validate, and optimize the model to maximize classification accuracy.
- Integrate the model with a Flask-based web application.
- Deploy a web interface for users to upload images and receive predictions.

2. Project Initialization and Planning Phase

2.1 Define Problem Statement

Manual blood cell classification is time-consuming, error-prone, and inaccessible in rural settings. An automated, accurate classification system is needed to support medical professionals in diagnosis and reduce delays.

2.2 Project Proposal (Proposed Solution)

The solution involves developing a blood cell classifier using MobileNetV2 and transfer learning. A web-based interface enables users to upload images and receive classification results instantly, reducing reliance on manual expertise.

2.3 Initial Project Planning

Initial planning included defining objectives, identifying the dataset, establishing the tech stack (TensorFlow, Flask, HTML), and structuring sprints for data, model, UI, and integration tasks.

3. Data Collection and Preprocessing Phase

3.1 Dataset Source

- **Source:** Kaggle Blood Cell Dataset
- **Classes:** Eosinophil, Lymphocyte, Monocyte, Neutrophil
- **Images:** 12,500 labeled JPEG images

3.2 Data Quality Report

- **Shape:** ~3000 images per class
- **Missing Data:** No missing data due to pre-cleaned dataset
- **Augmentation:** Images are already augmented and cropped

3.3 Data Exploration and Preprocessing

- Visual inspection and random sampling
- Normalization of pixel values
- Image resizing to match model input size
- Label encoding for multi-class classification

4. Model Development Phase

4.1 Feature Extraction with Transfer Learning

Used pre-trained **MobileNetV2** with frozen base layers, followed by:

- Flatten layer
- Dropout layer
- Dense Softmax output (4 neurons for 4 classes)

4.2 Training & Validation

- Optimizer: Adam
- Loss: Categorical Crossentropy

- Epochs: 5
- Achieved Accuracy: ~90% on validation data

5. Integration and Web App Development

5.1 Flask Web App

- **A Flask-based web application was developed to make the classification model accessible through a simple user interface.**
- **The application allows users to upload a blood cell image.**
- **The model processes the image and predicts the cell type (Eosinophil, Lymphocyte, Monocyte, or Neutrophil).**
- **The predicted class along with the confidence score is displayed on the result page.**

5.2 HTML Interface

- **home.html:**
    - **Provides a file input form for users to upload an image.**
    - **Contains a Predict button to trigger classification.**
- **result.html:**
    - **Displays the predicted blood cell type.**
    - **Shows the confidence/probability of prediction.**
    - **Offers a clean and simple interface for ease of use by medical professionals and students.**

6. Testing

6.1 Model Prediction

- The trained MobileNetV2 model was tested on **unseen blood cell images** from the test dataset.

- The model predicted the **correct blood cell class** (Eosinophil, Lymphocyte, Monocyte, Neutrophil) with high accuracy.

6.2 Accuracy Score

- The overall **accuracy** achieved on the test dataset is **89.3%**.
- Model accuracy was evaluated using **accuracy_score** from sklearn.

6.3 Classification Report

- A detailed classification report was generated including:
    - **Precision**
    - **Recall**
    - **F1-score**
    - **Support (number of images per class)**

Class	Precision	Recall	F1-Score	Support
Eosinophil	0.82	0.81	0.82	725
Lymphocyte	0.90	0.99	0.94	762
Monocyte	0.98	0.96	0.97	759
Neutrophil	0.87	0.80	0.83	742

6.4 Training and Validation Accuracy

- Accuracy increased steadily over 5 epochs.
- Final training accuracy: **~91%**
- Final validation accuracy: **~89%**
- The model showed **no overfitting** and performed well on both sets.

6.5 Confusion Matrix

- A confusion matrix was plotted to evaluate **class-wise prediction performance**.

- Most predictions are concentrated along the diagonal, indicating **correct classifications**.
- Minor confusion observed between Eosinophils and Neutrophils due to similar visual traits.

7. Advantages & Disadvantages

Advantages:

- Highly accurate due to transfer learning.
- Reduces diagnostic time and manual workload.
- Simple and intuitive web interface.
- Scalable for remote and rural healthcare integration.

Disadvantages:

- Currently supports only 4 blood cell types.
- Model performance depends on the quality of input images.
- Web app is local-only; deployment on cloud servers is pending.

8. Conclusion

HematoVision successfully automates blood cell classification using a transfer learning-based deep learning model. The system achieves high accuracy and integrates seamlessly into a Flask-based web interface, allowing real-time predictions. This solution enhances diagnostic speed, supports medical learning, and improves access to pathology expertise in underserved areas.

9. Future Scope

- Expand classification to include additional types such as **basophils** or **abnormal blood cells**.
- Increase dataset size and diversity for better generalization.
- Host the model and UI on the **cloud** for universal access in clinics and labs.

- Integrate **multilingual support** and options to **export results** as PDFs or reports.

10. Appendix

10.1 Source Code

- Model Notebook: model.py
- Flask Application: app.py

10.2 GitHub / Demo Link

(Add your GitHub repository or live deployment link here)