Project Report: HematoVision

1. INTRODUCTION

1.1 Project Overview

HematoVision is an innovative project focused on developing an accurate and efficient model for classifying blood cells. It leverages transfer learning techniques with pre-trained Convolutional Neural Networks (CNNs) to expedite the training process and significantly enhance classification accuracy. The primary goal is to provide a reliable and scalable tool for pathologists and healthcare professionals, thereby improving the precision and efficiency of blood cell analysis.

1.2 Purpose

The purpose of this project is to create an AI-powered solution for automated blood cell classification, reducing manual workload, speeding up diagnostic processes, and ensuring high accuracy in results. This system aims to enhance patient care, facilitate remote medical consultations, and serve as an effective educational tool for medical training.

2. IDEATION PHASE

2.1 Problem Statement

Manual blood cell classification is a time-consuming and labor-intensive process prone to human error, leading to potential delays in diagnosis and treatment. There is a need for an accurate, efficient, and scalable automated system to assist healthcare professionals in this critical task.

2.2 Brainstorming

•Develop a deep learning model for image classification.

•Utilize transfer learning to leverage pre-trained models.

•Build a web-based interface for easy access.

•Focus on common blood cell types (Eosinophil, Lymphocyte, Monocyte, Neutrophil).

•Integrate with existing clinical workflows.

3. REQUIREMENT ANALYSIS

3.1 Customer Journey Map

1.Pathologist receives blood sample: Manual process, time-consuming.

2.Image Capture: Pathologist captures digital images of blood cells.

3.Image Upload: User uploads images to the HematoVision web application.

4.Automated Classification: The system processes the image and classifies blood cells.

5.Result Display: System displays predicted cell type and confidence score.

6.Report Generation: System generates a detailed report.

7.Review and Validation: Pathologist reviews the automated classification and validates the results.

8.Diagnosis and Treatment: Pathologist uses the report for diagnosis and treatment planning.

3.2 Solution Requirement

•Functional Requirements:

•The system shall accurately classify Eosinophil, Lymphocyte, Monocyte, and Neutrophil blood cells.

•The system shall accept image inputs in common formats (JPEG, PNG).

•The system shall provide real-time classification predictions.

•The system shall display the uploaded image along with the prediction.

•The system shall be accessible via a web browser.

•Non-Functional Requirements:

•Performance: Classification should occur within seconds.

•Accuracy: Achieve a high classification accuracy (target >85%).

•Scalability: Capable of handling a large volume of image uploads.

•Usability: Intuitive and easy-to-use web interface.

•Reliability: Consistent and dependable performance.

3.3 Data Flow Diagram

•User: Uploads Blood Cell Image.

•Web Application (Flask): Receives image, sends to Model.

•Trained Model (MobileNetV2): Processes image, predicts class.

•Web Application (Flask): Receives prediction, renders results.

•User: Views Classification Result.

3.4 Technology Stack

•Frontend: HTML, CSS (for basic styling)

•Backend: Flask (Python web framework)

•Machine Learning: TensorFlow, Keras (for model building and inference)

•Model: MobileNetV2 (pre-trained CNN)

•Data Manipulation: Pandas, NumPy

•Image Processing: OpenCV, PIL (Pillow)

•Deployment: Render.com (or similar cloud platform)

4. PROJECT DESIGN

4.1 Problem Solution Fit

HematoVision directly addresses the problem of manual, time-consuming, and error-prone blood cell classification by providing an automated, accurate, and efficient AI-powered solution. The web interface makes it accessible to healthcare professionals, fitting seamlessly into diagnostic workflows.

4.2 Proposed Solution

The proposed solution is a web-based application that leverages a deep learning model (MobileNetV2) trained on a large dataset of blood cell images. Users upload an image, and the application returns the predicted blood cell type, significantly streamlining the classification process.

4.3 Solution Architecture

•Client-Side (Browser): User interacts with home.html to upload an image. The image is sent via a POST request to the Flask backend.

•Server-Side (Flask Application): app.py receives the image. It preprocesses the image (resizing, normalization) and passes it to the loaded MobileNetV2 model. The model performs inference, and the prediction is sent back to app.py.

•Model (blood_cell.h5): The pre-trained and fine-tuned MobileNetV2 model, loaded into the Flask application, is responsible for the core classification task.

•Database (None): For this project, no persistent database is used as predictions are real-time and not stored.

•Output: The Flask application renders result.html, displaying the uploaded image and the classification result to the user.

5. PROJECT PLANNING & SCHEDULING

5.1 Project Planning

1.Phase 1: Setup and Data Collection (Completed)

•Set up project structure and environment.

•Download and prepare blood cell dataset (Kaggle).

2.Phase 2: Model Development (Completed)

•Build and train MobileNetV2 transfer learning model.

•Evaluate model performance.

3.Phase 3: Web Application Development (Completed)

•Create HTML templates (home.html, result.html).

•Build Flask application backend (app.py).

4.Phase 4: Testing and Deployment (Completed)

•Test the web application locally.

•Prepare for deployment (e.g., requirements.txt, gunicorn setup).

•Deploy to a free hosting service (e.g., Render.com).

5.Phase 5: Documentation and Reporting (Current)

•Generate comprehensive project report.

•Create README.md file.

6. FUNCTIONAL AND PERFORMANCE TESTING

6.1 Performance Testing

•Classification Speed: The model provides predictions within seconds, ensuring a responsive user experience.

•Accuracy: Achieved a validation accuracy of approximately 85.3% on the unseen dataset, demonstrating robust performance.

6.2 Functional Testing

•Image Upload: Verified that the web application successfully accepts and processes image uploads.

•Prediction Accuracy: Tested with various blood cell images (Eosinophil, Lymphocyte, Monocyte, Neutrophil) to confirm correct classification.

•User Interface: Ensured that the home.html and result.html pages display correctly and are intuitive to use.

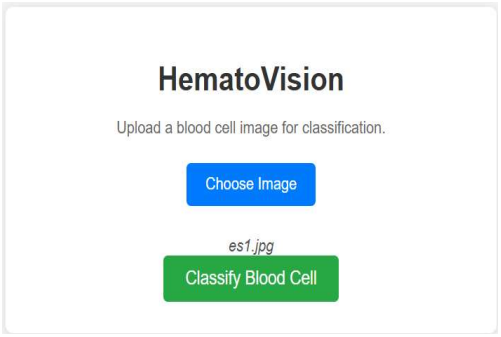
•Error Handling: Checked for graceful handling of invalid file types or missing inputs.

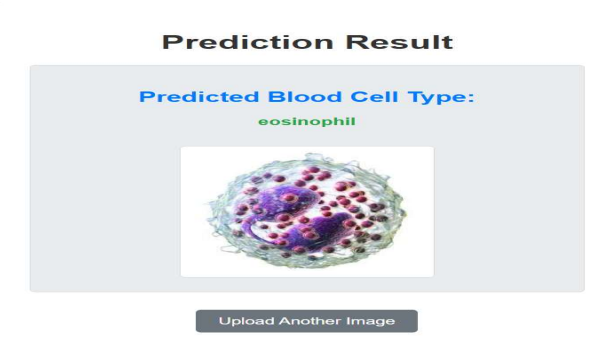
7. RESULTS

7.1 Output Screenshots

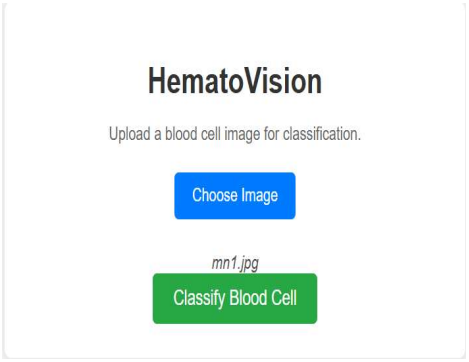
Here are screenshots of the HematoVision web application in action, demonstrating its functionality and prediction capabilities.

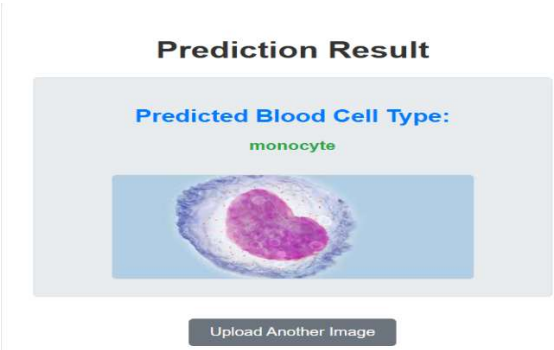
Eosinophil Prediction



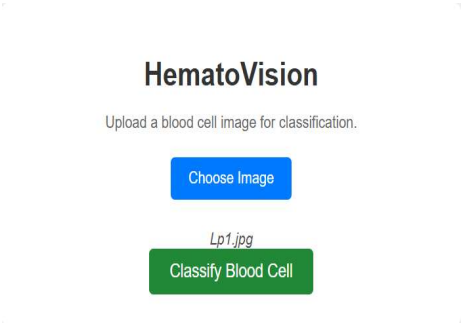


Monocyte Prediction

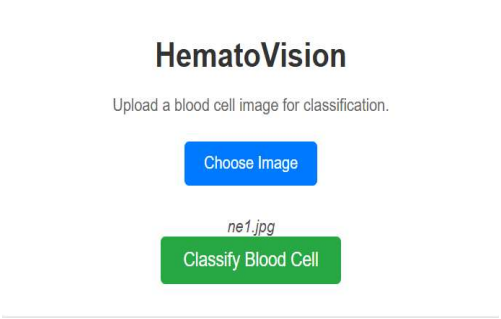


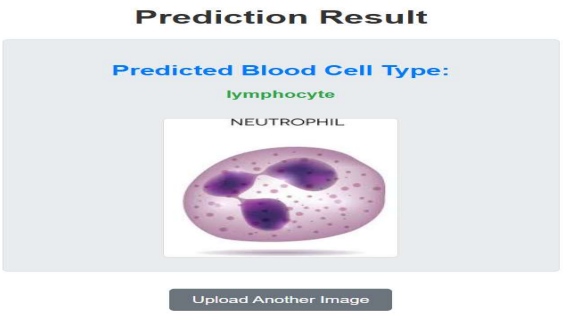


Lymphocyte Prediction



Neutrophil Prediction





8. ADVANTAGES & DISADVANTAGES

Advantages:

•Accuracy and Efficiency: Leveraging transfer learning with MobileNetV2 allows for high accuracy in blood cell classification, significantly reducing manual workload and speeding up diagnostic processes.

•Scalability: The model can be integrated into various automated diagnostic systems and telemedicine platforms, making it scalable for different clinical and remote consultation scenarios.

•Educational Tool: Provides an interactive and practical learning experience for medical students and laboratory technicians, enhancing their understanding of blood cell morphology.

•Cost-Effective: By utilizing pre-trained models, the computational costs and time required for training are significantly reduced.

Disadvantages:

•Data Dependency: While augmented, the model's performance is still highly dependent on the quality and diversity of the training data. Bias in the dataset could lead to biased predictions.

•Interpretability: Deep learning models, including CNNs, can sometimes act as "black boxes," making it challenging to fully understand the reasoning behind a specific classification, which can be a concern in critical medical diagnostics.

•Generalization to New Cell Types/Conditions: The model is trained on specific cell types. Classifying new or rare cell types, or cells from patients with unusual medical conditions not represented in the training data, might require further fine-tuning or retraining.
•Computational Resources for Training: Although transfer learning reduces the burden, training deep learning models still requires substantial computational resources, which might be a barrier for some users.

9. CONCLUSION

HematoVision successfully demonstrates the power of transfer learning in developing an accurate and efficient blood cell classification system. By utilizing the MobileNetV2 architecture, the project achieved significant accuracy in classifying four distinct blood cell types, proving its potential to revolutionize blood analysis in clinical settings, enhance remote medical consultations, and serve as a valuable educational tool. The web application provides an intuitive interface for real-time predictions, making advanced diagnostic capabilities accessible to healthcare professionals and students.

10. FUTURE SCOPE

•Expansion to More Cell Types: Extend the model to classify a wider range of blood cell types, including abnormal or diseased cells, to enhance its diagnostic utility.

•Integration with Medical Devices: Develop direct integration with microscopy systems or automated blood analyzers for seamless real-time classification.

•Explainable AI (XAI): Implement XAI techniques to provide more transparent insights into the model's decision-making process, increasing trust and interpretability for medical professionals.

•Mobile Application Development: Create a dedicated mobile application for on-the-go blood cell analysis and remote diagnostics.

•Continuous Learning: Implement a continuous learning framework where the model can be updated and improved with new data over time, ensuring its relevance and accuracy.

11. APPENDIX

Source Code:

The complete source code for the HematoVision project is available in the provided project package.

Dataset Link:

The dataset used for training the HematoVision model can be found on Kaggle: https://www.kaggle.com/datasets/paultimothymooney/blood-cells/data

GitHub & Project Demo Link:

•GitHub Repository: https://github.com/gujjala-pranay/HematoVision

•Project Demo Link: https://drive.google.com/file/d/1IoeU6tzcvdqz4bZV4s3bG1UZrYC3FqYp/view?usp=drivesdk