HematoVision: Advanced Blood Cell Classification Using Transfer Learning

Team ID: LTVIP2025TMID45379

Team Members:

- Akula Nithesh(Team Lead)
- Budigi Devendra Yadav
- Kathi Ramu
- Sirollapurna ChandraBalaji

Phase 1: Brainstorming and Ideation

Objective: To conceptualize a practical and efficient solution for automated classification of blood cells using deep learning methodologies, with a focus on medical relevance and real-time deployment potential.

Key Points:

- Problem Statement: Manual identification of blood cells is time-consuming, prone to errors, and requires expert pathologists.
- Proposed Solution: Develop a deep learning model using transfer learning to accurately classify different types of blood cells from microscopic images.
- Target Users: Medical professionals, diagnostic labs, healthcare researchers, and AI in healthcare developers.
- Expected Outcome: A web-based application capable of accurately classifying blood cells using an uploaded image, improving diagnostic accuracy and efficiency.

Phase 2: Requirement Analysis

Objective: To identify all technical and functional needs essential for the development and deployment of the blood cell classification system.

Key Points:

- Technical Requirements:
    - o Python 3.9 or 3.10
    - o TensorFlow >= 2.10

o Flask (for web app integration)

o NumPy, Pandas, Matplotlib, OpenCV, Seaborn

o Anaconda / Google Colab / VS Code (for development)

• Functional Requirements:

o User uploads a blood cell image

o The system preprocesses the image o The trained model predicts the cell type

o Result is displayed with classification confidence

• Constraints & Challenges:

o Model performance depends on dataset quality

o File compatibility for .h5 in different environments

o Limitations on Colab runtime and file storage

o Hardware limitations on local machines for training

Phase 3: Project Design

Objective: To establish a clear and scalable system architecture and define user flow for seamless application interaction.

• User Flow:

o User accesses the application (locally or via browser)

o Uploads image of blood cell

o Backend model processes and classifies the image

o Result is shown on the frontend

Phase 4: Project Planning

Objective: To outline the project timeline, task distribution, and dependencies.

Day 1

- **Member 1 will perform environment setup and package installation (3 hours) using Anaconda and Python to prepare the development environment.**
- **Member 2 will begin dataset collection (2 hours) by accessing and organizing the required blood cell image data.**

🗓️ **Day 2**

- **Member 2 will complete dataset preprocessing (2 hours) including cleaning and formatting the images.**
- **Member 3 will start model building using transfer learning (3 hours) with preprocessed data and TensorFlow.**

🗓️ **Day 3**

- **Member 3 will complete training and evaluation of the classification model (2 hours) to ensure good accuracy.**
- **Members 1 and 4 will begin Flask web app integration (2 hours) by preparing the frontend and backend structure.**

🗓️ **Day 4**

- **Members 1 and 4 will complete web app integration with the trained model (1 hour) to ensure full functionality.**
- **Members 2 and 3 will conduct system testing and debugging (2 hours) to identify and fix any bugs.**

🗓️ **Day 5**

- **The entire team will perform final testing, prepare the presentation, and deploy the application (3 hours) to make it demo-ready.**

Phase 5: Project Development

Objective: To build and test the system iteratively, addressing challenges and refining model performance.

Key Points:

• Technology Stack Used:

o Language: Python

o Frameworks: TensorFlow, Flask

o Tools: Google Colab, Anaconda, Jupyter Notebook, VS Code

• Development Process:

1. Data preprocessing and augmentation

2. Building and training model using transfer learning (e.g., MobileNetV2, ResNet50) 3. Model evaluation and tuning

4. Web application integration using Flask

5. Deployment and testing

• Challenges and Fixes:

o Issue: .h5 file not opening in VS Code

▪ Fix: Used correct Python environment with TensorFlow installed (Python 3.9/3.10)

o Issue: Jupyter notebook not launching from virtual environment

▪ Fix: Installed Jupyter inside the specific conda environment and added kernel

o Issue: TensorFlow install errors

▪ Fix: Switched to supported Python version (3.10) and used clean virtual environment