Detection of White Blood Cell Cancer Using Image Processing

Jayasri M[1], Azha Periasamy[2]

[1]Department of Electronics and Instrumentation, Bharathiar University, Coimbatore, Tamilnadu, India

[2]Department of Electronics and Instrumentation, Bharathiar University, Coimbatore, Tamilnadu, India

---***---

Abstract - This paper proposes an automatic system for detecting WBC cancers like AML, ALL, CLL, and CML, integrating image processing techniques. By automating analysis through advanced algorithms, it eliminates operator variability, ensuring consistent and objective results. Machine learning models classify cells for high accuracy, and seamless healthcare integration. This approach promises to enhance diagnostic efficiency and reliability in WBC cancer detection.

Key Words: Image processing; WBC cancer; Leukemia; CNN

1. INTRODUCTION

The research aims to develop and implement an automated system for detecting white blood cell (WBC) cancers, encompassing acute myeloid leukemia (AML), acute lymphoblastic leukemia (ALL), chronic lymphocytic leukemia (CLL), and chronic myeloid leukemia (CML). This system integrates advanced image processing techniques to overcome challenges associated with manual diagnostic methods, such as subjectivity, variability, and limited scalability. The primary objectives include designing an integrated system architecture with sophisticated algorithms for automated analysis of blood and bone marrow samples. Advanced image processing algorithms will be employed to accurately identify and classify various types of white blood cells, including cancerous ones, while machine learning models will be trained to enhance sensitivity and specificity in detection. Through rigorous testing and evaluation, this research endeavors to provide a robust, objective, and scalable solution for WBC cancer detection, ultimately improving diagnostic accuracy and patient outcomes.

2. MATERIALS AND METHODS

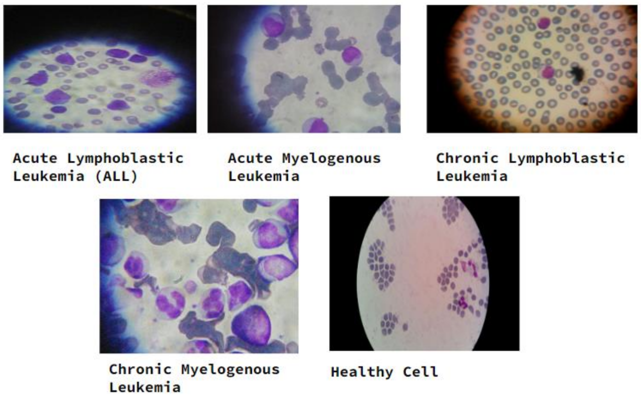
2.1 SOFTWARE AND TOOLBOX

MATLAB, developed by MathWorks, is a high-level programming language and interactive environment widely utilized in academia, industry, and research for numerical computing, data analysis, and visualization. Known for its focus on matrix manipulation and linear algebra operations, MATLAB provides an easy-to-learn programming language with a command-line interface and integrated development environment. Renowned for its powerful numerical computing capabilities, MATLAB offers extensive tools for data analysis and visualization, along with a vast ecosystem of toolboxes for specialized domains like image processing, control systems, and machine learning. Its versatility and ease of use make it a preferred choice for professionals and researchers worldwide, providing comprehensive support for algorithm development, simulation, and visualization across diverse application domains.

2.2 METHODOLOGY

2.2.1 Loading the training data

Before we train our convolutional neural network (CNN) model to detect blood cancer, we need a dataset of microscopic blood sample images. These images are typically labeled with the type of cancer present in the sample. We organize these images into a convenient format using the `imageDatastore` function, which creates an `ImageDatastore` object. This object stores the images along with their corresponding labels, making it easy to access and manage the dataset during training.


Acute Lymphoblastic Leukemia (ALL)
Acute Myelogenous Leukemia
Chronic Lymphoblastic Leukemia
Chronic Myelogenous Leukemia
Healthy Cell

(Fig 1: Sample of Input Images)

2.2.2 Pre-processing: global contrast stretching:

The quality of the input images can significantly impact the performance of our CNN model. To ensure consistent and optimal image quality, we apply a pre-processing technique called global contrast stretching. This technique adjusts the intensity values of the image to maximize the contrast between the darkest and lightest pixels. By stretching the intensity range, we enhance the visual details and improve the interpretability of features within the images.

2.2.3 Data augmentation:

To augment our training dataset and improve the robustness of our CNN model, we employ data augmentation techniques. These techniques involve generating additional training samples by applying transformations such as resizing, rotation, flipping, and translation to the original images. By introducing variations in the training data, data augmentation helps prevent overfitting and improves the model's ability to generalize to unseen data.

2.2.4 Defining the CNN architecture:

The architecture of our CNN model determines its structure and organization of layers. Each layer in the CNN performs specific operations on the input data, such as feature extraction, non-linearity, and classification. Our CNN architecture comprises several types of layers, including convolutional layers for feature extraction, max-pooling layers for spatial downsampling, and fully connected layers for classification. The final output layer uses a softmax activation function to produce a probability distribution over the possible classes.

2.2.5 Setting training options:

Training a Convolutional Neural Network (CNN) entails adjusting its parameters, such as weights and biases, to minimize a predetermined loss function. We specify various training options to control the training process, such as the choice of optimization algorithm (e.g., stochastic gradient descent), the number of training epochs (iterations), the mini-batch size (number of samples processed in each iteration), and the learning rate (the rate at which the model parameters are updated). Additionally, we specify options for shuffling the training data and displaying training progress.

2.2.6 Training the CNN:

With the training data and architecture in place, we proceed to train the CNN model. During training, the model learns to map input images to their corresponding labels by iteratively adjusting its parameters using backpropagation and gradient descent. The training process involves feeding batches of training samples into the model, computing the loss between the predicted and true labels, and updating the model parameters to minimize the loss. This iterative process continues for multiple epochs until the model converges to a satisfactory level of performance.

2.2.7 Saving the trained network:

Once training is complete, we save the trained CNN model to disk for future use. Saving the model allows us to reuse it for inference tasks, where new blood sample images can be classified without the need for retraining. The trained model is typically serialized and stored in a format such as a MATLAB `.mat` file, preserving its architecture and parameters.

2.2.8 Loading the trained network:

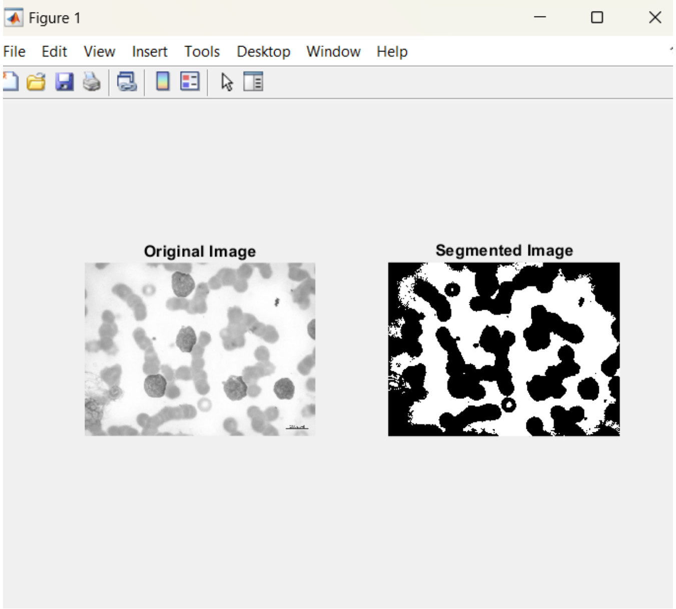
To utilize the trained CNN model for inference, we load the saved model from the disk into memory. This step involves deserializing the model file using the `load` function, restoring the model's architecture and parameters to their trained state.

2.2.9 Processing the input image:

When a new blood sample image is provided for classification, it undergoes pre-processing steps similar to those applied during training. Pre-processing prepares the input image for feature extraction and classification by enhancing its visual quality and reducing noise or artifacts that may affect the accuracy of the classification.

2.2.10 Segmentation:

Segmentation involves dividing an image into distinct and interpretable regions or objects. In our approach, we employ dual threshold segmentation, which divides the image into foreground (object) and background regions based on intensity thresholds. This step isolates the relevant components of the blood sample image for subsequent analysis and feature extraction.



(Figure: Segmented Image)

2.2.11 Feature extraction:

Feature extraction involves identifying and extracting informative characteristics or patterns from the segmented image. In our workflow, we perform binarization, morphological erosion, and median filtering as feature extraction techniques. These operations highlight relevant structures and attributes of the blood sample image, aiding in the subsequent classification process.

2.2.12 Resizing the image:

The processed image is resized to match the input dimensions expected by the CNN model. Resizing standardizes the size of input images, ensuring compatibility with the CNN architecture and enabling consistent processing across different image resolutions.

2.2.13 Classification using CNN:

The resized image is inputted to the trained CNN model for classification. CNN leverages its learned parameters to extract hierarchical features from the input image and predict the blood cancer type. The final classification decision is based on the output of the of the softmax layer.

3. RESULTS

Image preprocessing techniques, including noise reduction, contrast enhancement, and normalization, significantly improved the quality of WBC images, enhancing subsequent analysis. Various feature extraction methods captured relevant information from WBC images, facilitating effective cancer detection. Machine learning algorithms trained on these features demonstrated the effectiveness of the proposed approach in distinguishing between normal and cancerous WBCs.

4. CONCLUSIONS

In conclusion, this paper has presented a comprehensive framework for WBC cancer detection using image processing techniques. Through preprocessing, feature extraction, and classification model development, the system demonstrated high efficiency in distinguishing between normal and cancerous WBCs. Overall, this research contributes to advancing healthcare technologies by leveraging interdisciplinary approaches to improve cancer detection and diagnosis.

5. REFERENCES

[1] Lorenzo Putzu, and Cecilia Di Ruberto. White Blood Cells Identification and Counting from Microscopic Blood Image . World Academy of Science, Engineering and Technology International Journal of Medical, Health, Biomedical and Pharmaceutical Engineering Vol:7, No:1, 2013

[2] YanLi, Rui Zhu, Lei Mi, Yihui Cao, and Di Yao . Segmentation of White Blood Cell from Acute Lymphoblastic Leukemia Images Using Dual-Threshold Method. Hindawi Publishing Corporation Computational and Mathematical Methods in Medicine Volume 2016.

[3] Savita Dumyan, Ankush Gupta. An Enhanced Technique for Lymphoblastic Cancer Detection Using Artificial Neural Network. International Journal of Advanced Research in Computer Science and Electronics Engineering Volume 6, Issue 3, March 2017.

[4] Sos Agaian, Monica Madhukar, and Anthony T. Chronopoulos. Automated Screening System for Acute Myelogenous Leukemia Detection in Blood Microscopic Images. IEEE SYSTEMS JOURNAL,VOL.8,NO.3,SEPTEMBER 2014 995.

[5] Dr.T.Karthikeyan, N.Poornima. Microscopic Image Segmentation Using Fuzzy C Means For Leukemia Diagnosis. International Journal of Advanced Research in Science, Engineering and Technology Vol. 4, Issue 1 , January 2017.

[6] Sarmad Shafique and Samabia Tehsin. Computer-Aided Diagnosis of Acute Lymphoblastic Leukaemia. Hindawi Computational and Mathematical Methods in Medicine Volume 2018.

[7] Karar, M.E.; Alotaibi, B.; Alotaibi, M. Intelligent Medical IoT-Enabled Automated Microscopic Image Diagnosis of Acute Blood Cancers. Sensors 2022, 22, 2348