Implementation of Convolutional Neural Networks for the Purpose of Five Types of White Blood Cells Automatic Counting

Grzegorz Dralus[1][0000−0001−7963−8602], Damian Mazur[1][0000−0002−3247−5903], Konrad Lukiewicz[1][0000−0003−2463−9338], Michal Podpora[2][0000−0002−1080−6767], Jacek Bartman[3][0000−0001−7372−2029], Henryk Racheniuk[4][0000−0001−7372−2029], Tomasz Kajdanowicz[5][0000−0002−6688−5505], and Aleksandra Kawala-Sterniuk[5][0000−0001−7826−1292]

[1] Department of Electrical and Computer Engineering Fundamentals, Rzeszow University of Technology, 35-959 Rzeszow, Poland
{gregor,mazur}@prz.edu.pl, kondluki@wp.pl
[2] Institute of Computer Science, University of Opole, 45-052 Opole, Poland
michal.podpora@uni.opole.pl
[3] Institute of Computer Science, University of Rzeszow, 35-310 Rzeszow, Poland
jbartman@ur.edu.pl
[4] Faculty of Physical Education and Physiotherapy, Opole University of Technology, 45-753 Opole, Poland, h.racheniuk@po.edu.pl
[5] Department of Artificial Intelligence, Faculty of Information and Communication Technology, Wroclaw University of Science and Technology, 50-370 Wroclaw, Poland
{tomasz.kajdanowicz,aleksandra.kawala-sterniuk}@pwr.edu.pl

Abstract. This study proposes a deep Convolutional Neural Network (CNN) for automated recognition and classification of five WBC types from microscopic images. Various network structures, filter sizes, numbers of hidden layers, and different learning algorithms were evaluated to achieve high accuracy. In this way, 18 network variants, including different learning algorithms, were tested. Several of them achieved very high accuracy in recognizing and scoring 5 types of WBCs. The efficiency of the proposed models can be used to help medical professionals, offering potential support in enhancing diagnostic efficiency and blood analysis.

Keywords: Deep Learning · Convolutional Neural Networks · White Blood Cell Classification · Automated Blood Cell Counting · Blood Smear Analysis · Leukocyte Detection · White Blood Cell Classification · Hematological Diagnostics

1 Introduction

The blood system delivers oxygen and nutrients while transporting immune cells, mainly white blood cells (WBCs), which originate in the hematopoietic system [7, 12]. WBCs circulate in the blood and lymphatic system, playing key roles in immune defense, inflammation, allergic responses, and cancer protection. Their

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97554-7_29

differentiation helps assess cell proportions, deficiencies, excesses, and atypical forms [7].

WBCs are classified as granulocytes (neutrophils, eosinophils, basophils) with segmented nuclei and agranulocytes (lymphocytes, monocytes) with spherical nuclei [46]. Traditional smear analysis is time-consuming and operator-dependent [5], while automated systems offer higher accuracy and speed [48]. This study develops and evaluates CNN-based models for classifying five WBC types.

Several automated leukocyte classification systems exist [45, 6], but their high cost is due to complex acquisition and classification processes. They rely on segmentation, feature extraction, and pattern recognition, with segmentation strongly affecting accuracy [32, 24, 3, 19].

Feature extraction also influences performance [35, 33]. Deep learning, particularly CNNs, overcomes these limitations and excels in image classification [39, 10, 20, 26].

For the purpose of image recognition numerous standard structures of these models known as, for example, Alexnet [21], GoogleNet [37], ResNet [22], YOLO [17] and many others, as well as designing their own structures for this purpose are being applied [30, 2]. Many algorithms have been used for leukocyte classification, including artificial neural networks (ANN) [25, 36], support vector machine (SVM) [28], or naive Bayes classifier [27].

High classification performance depends on image processing, segmentation, and feature selection, especially for shallow models. Some studies focus on nucleus segmentation, others include both nucleus and cytoplasm. Cao et al. [43] used component filtering for fast nucleus segmentation, while [42] applied saturation analysis with OTSU thresholding.

Na Dong et al. developed an adaptive segmentation method for unevenly lit blood smears, using CART and PSO-SVM for high-accuracy WBC classification [8]. Advances in AI, including CNNs, support early disease diagnosis. Luis et al. applied CNN-based transfer learning with SVM for leukemia detection [38], while Qin et al. used a deep residual network to classify 40 cell types [29].

A classification system for six types of WBCs, including abnormal cells, using data augmentation techniques in a deep learning approach with CNNs was presented [14].

Shahin et al. used CNNs for WBC classification, comparing AlexNet (91.2% accuracy) and LeNet-5, with their custom CNN achieving 96% accuracy [34]. Khan et al. employed AlexNet-based CNNs with feature selection (FS) and Extreme Learning Machine (ELM), developing MLANET-FS-ELM, which achieved high accuracy [17]. Wang et al. [41] proposed a CNN-based WBC classification system without segmentation, treating leukocyte recognition as object detection using SSD and YOLOv3, identifying 11 WBC categories with high accuracy. Acevedo et al. [1] trained VGG-16 and InceptionV3 CNNs for classifying eight blood cell types, including five WBC classes. Their models used transfer learning with SVM classification or direct fine-tuning on peripheral blood images for comprehensive classification.

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97554-7_29

2 CNNs and Learning Techniques

Convolutional Neural Networks (CNNs) are widely used for image and speech recognition, effectively handling nonlinear patterns. They mimic human vision by automatically extracting key features, replacing the need for manual, less accurate methods [11].

Convolution layers identify object features in an image by applying filters of specified sizes (e.g., 3×3, 5×5, 7×7) to scan and process pixels. Each surrounding pixel contributes with a weight, stored in a mask, influencing the new pixel value. Filters consider neighboring values to enhance feature extraction, with odd-sized masks ensuring a central reference pixel. The mathematical definition of convolution filters is given with the below equation (1) [13]:

$$output_{[i][j]} = \sum input_{[a][b]} * filter_{[i-a][j-b]}, \tag{1}$$

where:
$output$ – filtered image, $input$ – original image,
$filter$ – convolution kernel applied to the input image,
i and j – rows and columns of the output image,
a and b – rows and columns of the input image.

Formula (2) for calculating the size of the convolution layer:

$$W_{out} = \frac{W_{in} - F + 2P}{S} + 1 \quad , \quad H_{out} = \frac{H_{in} - F + 2P}{S} + 1, \tag{2}$$

where:
W_{in} – the height of the image,
H_{in} – the width of the image,
F – the width and height of the filter,
S – the convolution stride,
P – the padding.

The output size is calculated for each channel separately.

Pooling layers reduce the feature map size by combining neighboring pixels, typically using max or average pooling within a square matrix. This makes representations invariant to small changes, crucial for object recognition, where location matters less than presence. Average Pooling, applied after convolution layers, reduces image size while enhancing robustness to scale changes and shifts, allowing the network to focus on essential features rather than minor details [4].

The **Dropout layer** randomly deactivates some neurons during training to prevent overfitting, ensuring that the model does not rely too much on initial training samples [40].

The **Dense layer** (fully connected layer) links each neuron to each neuron in the next layer through individual weights. It transforms data representations, which is essential for complex tasks like image classification, fashion recognition, and NLP. Often used as the final layer, it maps the input data to the network output. The complete connection layer formulas are provided with the equation (3) below [40, 16]:

$$y_i = \rho(W_1 x_1 + ... + W_m x_m). \tag{3}$$

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97554-7_29

The **loss function** results from the softmax function in the classifier. In the classification layer, the loss function on the values of the softmax function assigns each input white blood cell to one of the mutually exclusive classes using the cross-entropy function for the coding scheme (4) [15]:

$$J(\theta) = -\frac{1}{N}\sum_{n=1}^{N}\sum_{i=1}^{K}\theta_i t_{ni}\ln\left(y_{ni}\right), \tag{4}$$

where:
N is the number of samples,
K is the number of classes,
θ_i is the class weight,
t_{ni} is an indicator that the nth sample belongs to the ith class,
y_{ni} is the output of sample n from class i, which in this case is the value of the softmax function.

2.1 Optimization Strategies in CNN Training

Gradient descent is a way to minimize the objective function $J(\theta)$ of the model parameters $\theta \in$ Rd by updating the parameters in the opposite direction of the gradient of the objective function $\nabla\theta J(\theta)$ due to the parameters. The learning rate η determines the size of the steps we take to reach a (local) minimum. The three gradient descent variants differ in data usage for gradient calculation, balancing update accuracy, and computation time.

The standard **Batch Gradient Descent** (BGD) calculates the gradient of the cost function against the θ parameters for the entire training data set (5):

$$\theta = \theta - \eta\nabla\theta J(\theta). \tag{5}$$

Batch gradient descent is slow as it computes gradients for the entire dataset per update, making it memory-intensive and unsuitable for online updates. **Stochastic Gradient Descent** (SGD) calculates the parameter gradient using only one training example, for example, for a pair of $x^{(i)}$ and $y^{(i)}$ from the training set expressed with the (6):

$$\theta = \theta - \eta\nabla\theta J(\theta; x(i), y(i)), \tag{6}$$

Adadelta [47] is an extension of Adagrad [44] that aims to reduce its aggressive, monotonically decreasing learning rate. Instead of accumulating all past quadratic gradients, Adadelta limits the window of accumulated past gradients to a fixed size w. The moving average $E[g^2]t$ at time $t$then depends only on the previous average and the current gradient (7):

$$E\left[g^2\right]_t = \gamma E\left[g^2\right]_{t-1} + (1-\gamma)g_t^2 \tag{7}$$

We set the γ to a similar value as the momentum term, around 0.9. Parameter vector update formulas $\Delta\theta_t$ (8) and (8):

$$\Delta\theta_t = -\eta \cdot g_t, \tag{8}$$

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97554-7_29

$$\theta_{t+1} = \theta_t + \Delta\theta_t \tag{9}$$

Adadelta's algorithm in the denominator includes the distribution of averages in previous quadratic gradients $E[g^2]t$ (see: (10)):

$$\Delta\theta_t = -\frac{\eta}{\sqrt{E[g^2]_t + \epsilon}} \cdot g_t. \tag{10}$$

The **RMSProp** algorithm [49] also has an adaptive learning rate. It was developed from the need to solve the radically decreasing learning rates of Adagrad. The RMSProp algorithm is, in fact, identical to the first Adadelta update vector described above (11) and (12):

$$E[g^2]_t = 0.9E[g^2]_{t-1} + 0.1g_t^2, \tag{11}$$

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{E[g^2]_t + \epsilon}} \cdot g_t. \tag{12}$$

RMSProp adjusts the learning rate by dividing it by the exponentially decaying mean of squared gradients, with recommended values $\gamma = 0.9$ and $\eta = 0.001$ [49].

The **ADAM** (Adaptive Moment Estimation) algorithm [18] computes adaptive learning rates by tracking both the average of past gradients and squared gradients, similar to momentum-based methods. The first and second moment estimates, m_t and v_t, are calculated as follows (13):

$$m_t = \beta_1 m_{t-1} + (1-\beta_1)g_t \quad , \quad v_t = \beta_2 m_{t-1} + (1-\beta_2)g_t^2 \tag{13}$$

Since m_t and v_t tend to tend toward zero, first and second moment estimates are calculated to counteract this, and then to update the parameters (14) and (15):

$$\widehat{m}_t = \frac{m_t}{1-\beta_1^t} \quad , \quad \widehat{v}_t = \frac{v_t}{1-\beta_2^t} \tag{14}$$

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{\widehat{v}_t} + \epsilon} \cdot \widehat{m}_t \tag{15}$$

The authors suggest default values of 0.9 for β_1 , 0.999 for β_2 , and 10^{-8} for ϵ .

The **Adamax** algorithm used the possibility of replacing the L_2 norm with the L_∞ norm, which generally also shows stable behavior. For this reason, the Adamax authors propose and show that the v_t factor of L_∞ converges to a stable value. To distinguish this algorithm from the Adam, u_t is used to denote v_t bounded by the infinity norm (16) [18]:

$$u_t = \beta_2^\infty v_{t-1} + (1-\beta_2^\infty)\,|g_t|^\infty = \max\left(\beta_2 \cdot v_{t-1}, |g_t|\right) \tag{16}$$

If we replace the expression $\sqrt{\widehat{v}_t} + \epsilon$ with u_t we get the Adamax update rule (17):

$$\theta_{t+1} = \theta_t - \frac{\eta}{u_t} \cdot \widehat{m}_t. \tag{17}$$

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97554-7_29

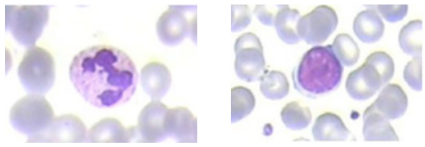
It should be added that u_t, based on the MAX operation, does not go quickly to zero is like m_t and v_t in ADAM. Default values of the algorithm $\eta = 0.002$, $\beta 1 = 0.9$ and $\beta 2 = 0.999$. The Nesterov-accelerated Adaptive Moment Estimation (**Nadam**) [9] extends Adam by incorporating Nesterov momentum, improving optimization performance. By combining Adam's equations with Nesterov momentum and applying simplifications, the final Nadam update rule is derived with the (18):

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{\widehat{v}_t} + \epsilon} \cdot \left(\beta_1 \widehat{m}_t + \frac{(1-\beta_1)}{1-\beta_1^t} g_t \right) \tag{18}$$

The algorithm has Nesterov momentum, which replaces the bias-corrected estimate of the momentum vector from the previous time step $\widehat{m}_{t-1}$ with the error-corrected estimate of the current momentum vector $\widehat{m}_t$.

3 Medical aspect of the work

White blood cells, essential to the immune system, are classified as granulocytes or agranulocytes based on cytoplasmic granules. Neutrophils fight pathogens [23], eosinophils target parasites, and basophils trigger inflammation. Monocytes indicate chronic inflammation, while lymphocytes increase during viral infections (see Fig. 1).



(a) (b)

Fig. 1. Cells: Neutrophil (a), Lymphocyte (b).

4 CNN Architectures and Model Variants

Classification accuracy was used as the primary metric, calculated as the ratio of correctly identified cells to actual cells. Gradient updates were based on error function results.

A total of 18 models were designed, varying in architecture, hidden layers, and filter sizes (Tables 1 and 2). Some models share a fixed structure, differing only in learning algorithms.

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97554-7_29

Table 1. Structure of the first 9 CNN Models.

	Model								
Layers	**I**	**II**	**III**	**IV**	**V**	**VI**	**VII**	**VIII**	**IX**
Conv2	32,3x3	32,3x3	32,3x3	32,3x3	32,3x3	32,3x3	32,3x3	32,3x3	32,3x3
Conv2	-	-	-	32,3x3	-	-	-	-	-
MaxP	2x2	2x2	2x2	2x2	2x2	2x2	2x2	2x2	2x2
Conv2	32,3x3	32,3x3	32,3x3	32,3x3	32,3x3	32,3x3	32,3x3	32,3x3	32,3x3
MaxP	2x2	2x2	2x2	2x2	2x2	2x2	2x2	2x2	2x2
Conv2	-	32,3x3	32,3x3	32,3x3	32,3x3	32,3x3	32,3x3	32,3x3	32,3x3
MaxP	-	2x2	2x2	2x2	2x2	2x2	2x2	2x2	2x2
Conv2	-	-	32,3x3	32,3x3	-	-	-	-	-
MaxP	-	-	2x2	2x2	-	-	-	-	-
Flatten	YES	YES	YES	YES	YES	YES	YES	YES	YES
Dense	32	32	32	32	32	32	32	32	32
Optim.	Adadelta	Adadelta	Adadelta	Adadelta	Adagrad	Adam	Adamax	Nadam	RMSprop

Table 2. Structure of the new 9 CNN Models.

	Model								
Layers	**X**	**XI**	**XII**	**XIII**	**XIV**	**XV**	**XIV**	**XVII**	**XVIII**
Conv2	32,3x3	16,3x3	64,3x3	128,3x3	32,3x3	32,5x5	32,3x3	32,3x3	32,3x3
Conv2	-	-	-	-	-	-	-	-	-
MaxP	2x2	2x2	2x2	2x2	2x2	2x2	3x3	2x2	2x2
Conv2	32,3x3	16,3x3	64,3x3	128,3x3	64,3x3	32,5x5	32,3x3	32,3x3	32,3x3
MaxP	2x2	2x2	2x2	2x2	2x2	2x2	3x3	2x2	2x2
Conv2	32,3x3	16,3x3	64,3x3	128,3x3	128,3x3	32,5x5	32,3x3	32,3x3	32,3x3
MaxP	2x2	2x2	2x2	2x2	2x2	2x2	3x3	2x2	2x2
Conv2	-	-	-	-	-	-	-	-	-
MaxP	-	-	-	-	-	-	-	-	-
Flatten	YES	YES	YES	YES	YES	YES	YES	YES	YES
Dense	32	32	32	32	32	32	32	32	32
Dropout	0.5	0.5	0.5	0.5	0.5	0.5	0.5	-	-
Dense	5	5	5	5	5	5	5	5	32
Dense	-	-	-	-	-	-	-	-	5
Optim.	SGD	Adadelta	Adadelta	Adadelta	Adadelta	Adadelta	Adadelta	Adadelta	Adadelta

4.1 Implementation and Data Preprocessing

The CNN model implementation for training and testing was written in Python, using TensorFlow, Keras, SciPy, and Scikit-Learn. The WBC classes (basophil, eosinophil, lymphocyte, monocyte, neutrophil) were assigned numerical labels $[0, 1, 2, 3, 4]$ for automated recognition and counting.

Images were taken from the LISC database [31] and preprocessed by resizing, splitting, and rotating each image by 5° increments to generate 72 variants. Manually filtered images with cropped cells were removed to avoid classification errors.

To speed up training, image sizes were reduced from 720×576 to 180×144 pixels. The dataset was split into $13,000$ training images ($2,600$ per class) and $3,250$ test images (650 per class).

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97554-7_29

4.2 Model evaluation

Model verification involved (1) monitoring error and accuracy during training, and (2) final evaluation on test data to confirm effectiveness. This process was also implemented in the application. Precision and Recall measure the ratio of correctly classified samples to all predicted and actual class samples, respectively (19):

$$Precision = \frac{TP}{TP + FP} \quad , \quad Recall = \frac{TP}{TP + FN} \tag{19}$$

The $F1$-Score is weighted average of Precision and Recall, and the classification accuracy is the number of properly recognized cells in relation to the number of all cells (20) and (21):

$$F1Score = \frac{2 * precision * recall}{precision + recall} \tag{20}$$

$$Accuracy = \frac{TP + TN}{TP + FN + TN + FN} 100\% \tag{21}$$

5 Model Performance and Accuracy

Table 3 shows the classification accuracy and classification loss for the learning and testing sets. Depending on the parameter changes applied, different accuracy and loss values were obtained.

Table 3. Classification Accuracy and Loss of CNN Models.

	Learning		**Testing**	
Model	**Acc (%)**	**Loss (%)**	**Acc (%)**	**Loss (%)**
I	94.50	0.1630	95.66	0.1296
II	96.56	0.1069	97.20	0.0825
III	97.36	0.0891	98.03	0.0426
IV	97.99	0.0785	97.38	0.0695
V	95.88	0.1243	96.06	0.1116
VI	79.86	0.4846	86.83	0.3214
VII	94.45	0.1283	98.12	0.0659
VIII	19.95	1.609	20.00	1.609
IX	92.89	0.3747	95.91	0.1223
X	79.77	0.5515	83.54	0.4875
XI	94.36	0.1682	96.25	0.1051
XII	96.29	0.1077	86.40	0.5160
XIII	78.81	0.4922	71.20	0.7710
XIV	98.32	0.0791	98.06	0.0522
XV	96.22	0.1696	98.37	0.0487
XVI	93.33	0.2072	97.85	0.0874
XVII	98.43	0.0680	97.48	0.0703
XVIII	99.53	0.0208	98.03	0.0732

Model I consists of two Conv2D and two MaxPooling layers, followed by Flatten, Dense, and Dropout layers, achieving 94.50% accuracy on training data and 95.66% on test data.

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97554-7_29

Model II, an extended version of **Model I** with three Conv2D-MaxPooling pairs, improves accuracy to 96.56% (training) and 97.20% (testing).

Model III includes four Conv2D-MaxPooling pairs, further enhancing accuracy to 97.36% (training) and 98.03% (testing), with classification losses below 0.1.

Model IV features five convolutional and four max pooling layers, with modifications in pooling placement. It achieves 97.99% accuracy on training and 97.35% on test data.

All models **I**–**IV** use Adadelta as the optimizer.

Model V shares **Model II**'s structure but uses Adagrad, resulting in lower accuracy since Adadelta (**Model II**) is an improved version of Adagrad. **Model VI** (same as **Model II**) uses Adam, reducing accuracy to 79.86% (training) and 86.83% (testing). **Model VII** (same as **Model II**) uses Adamax, improving classification to 94.45% (training) and 98.12% (testing), outperforming Adam. **Model VIII** (same as **Model II**) uses Nadam, but performs worse than **Model VI** (Adam), as Nadam is derived from it. **Model IX** (same as **Model II**) uses RMSprop, achieving 92.89% (training) and 95.91% (testing) accuracy. **Model X** (similar to **Model II**) with SGD results in lower accuracy. Among **Models V**–**X**, Adadelta (**Model II**) provided the best accuracy, except for **Model VII** (Adamax), which performed similarly. Future models will use Adadelta. **Model XI** (based on **Model II**) reduces Conv2D filters from 32 to 16, lowering accuracy to 94.36% (training) and 96.25% (testing). **Model XII** (same as **Model II**) increases filters to 64, achieving 96.29% (training) but only 86.4% (testing), suggesting overfitting. **Model XIII** (based on **Model II**) increased Conv2D filters to 128, leading to overfitting with 87.25% accuracy (training) but only 55.94% (testing), likely due to insufficient training time. **Model XIV** progressively increased filters ($32 \rightarrow 64 \rightarrow 128$) across layers, achieving 98.32% (training) and 98.06% (testing), showing strong performance. **Model XV** increased filter size from 3×3 to 5×5, improving accuracy to 96.22% (training) and 98.37% (testing). However, training time doubled, increasing computational costs. **Model XVI** increased max pooling filter size ($2 \times 2 \rightarrow 3 \times 3$), decreasing training accuracy (93.33%) but improving test accuracy (97.85%). **Model XVII** (without Dropout) achieved 98.43% (training) and 97.48% (testing), as Dropout artificially hinders learning but is inactive during testing. **Model XVIII** removed Dropout and added an extra Dense layer (32 neurons), enhancing feature learning. It achieved the highest accuracy: 99.53% (training) and 98.02% (testing), improving classification precision.

5.1 Model Comparisons and Interpretations

This study evaluates the impact of model structure, filter size/number, and optimizer choice on classification accuracy, using Model II as the baseline. Additional Conv2D and MaxPooling layers improved accuracy (**Model III**) but excessive layers reduced performance (**Model IV**). Removing Dropout and adding a Dense layer increased accuracy (**Model XVIII**).

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97554-7_29

Reducing filters lowered accuracy, while gradually increasing them across layers (**Model XIV**) matched **Model III**'s performance. Larger convolution filters improved test accuracy (**Model XV**), while larger max pooling filters slightly improved test accuracy but reduced training accuracy.

Most optimizers performed worse than Adadelta, except Adamax, which improved test accuracy but reduced training accuracy (**Model VII**). Adadelta was optimal, especially in **Models III**, **XIV**, and **XVIII**, which used fewer convolutional and max pooling layers than architectures like ResNet101, VGG16, and MobileNetV2.

Fig. 2 presents the best classification accuracies from all models across three test runs.

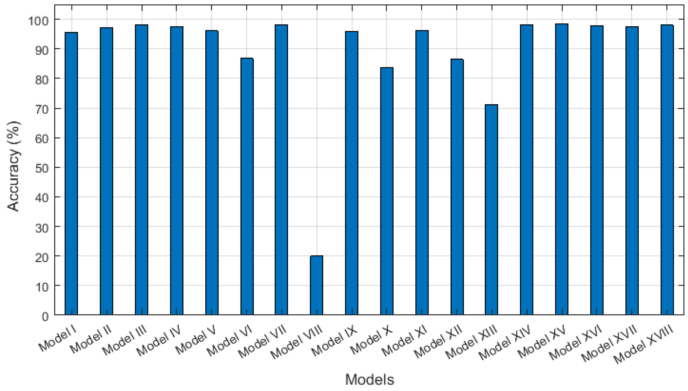


Fig. 2. Classification accuracy for the test set for all models.

The highest classification accuracy for the learning data was achieved by **Model XVIII** but for the test data by **Model XV**. However, in **Model XV**, the accuracy for test data is more than two percentage points higher than for learning data, not an anomaly in a sense, so **Model XV** will not be considered the best. In addition to these two, **Models III** and **XIV** achieved very high levels of WBC classification accuracy for both training and testing data. Analyzing the results obtained, it can be seen that in some models the classification accuracy for the test set is higher than for the training set.

These differences are generally small and may result from random errors, initial parameter values, optimization strategy, or the fixed 20-epoch training limit. Since most models show higher accuracy on training than test data, the dataset split appears appropriate and representative, supporting the validity of the results.

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97554-7_29

5.2 Validation and Model Generalization

The best-trained models can be used to identify and count WBCs. Based on classification accuracy and loss, the top models are **III**, **XIV**, and **XVIII**, with **Model III** chosen for detailed testing. This section presents its classification results for five WBC classes using a confusion matrix (Table 4) and a text report (Table 5), showing Precision, Recall, and $F1$-score.

Model III classifies Basophils and Eosinophils with the highest precision, while Lymphocytes and Neutrophils have the highest Recall. Overall, Neutrophils are identified most accurately ($F1$-score). The Macro avg. represents the average of all metrics (Table 5). The main classification error occurs with Monocytes, where 613 were correctly identified, but 37 were misclassified as Lymphocytes.

Table 4. Detailed Confusion Matrix for trained Model III.

WBC's	**True/Target class**				
Predicted	Baso	Eosi	Lymp	Mono	Neut
Basophil	636	0	0	0	0
Eosinophile	0	638	0	0	0
Lymphocyte	0	1	649	37	0
Monocyte	13	8	1	613	0
Neutrophil	1	3	0	0	650

Table 5. Textual classification report of Model III.

WBC's	**Precision**	**Recall**	**F1-score**	**Support**
Basophil	1.00	0.98	0.99	650
Eosinophile	1.00	0.98	0.99	650
Lymphocyte	0.94	1.00	0.97	650
Monocyte	0.97	0.94	0.95	650
Neutrophil	0.99	1.00	1.00	650
Macro avg	0.98	0.98	0.98	3250

Granulocytes are easier to classify due to their lobed nuclei, while agranulocytes (Monocytes and Lymphocytes) have less distinct features, making classification harder. Monocytes typically have kidney-shaped nuclei, but similarities with lymphocytes' oval nuclei can cause confusion. Model performance is evaluated using a confusion matrix (Table 4), which provides the basis for calculating classification metrics and identifying errors, with cell labels defined in Table 3.

6 Summary and Discussion

To evaluate our approach, we used standard accuracy metrics and compared results with previous WBC classification studies. Some methods targeted more

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97554-7_29

Table 6. Comparison of classification results of five types of WBCs with selected method in terms of accuracy.

Author	Method	Baso	Eosi	Lymp	Mono	Neut	mean	number
Rezatofighi	SVM+LBP	89.69	100.00	93.10	95.83	96.43	96.20	251
Wang	YOLOv3	99.30	99.90	93.70	91.20	95.50	95.92	1120
Reena	AlexNet	100.00	100.00	98.87	99.24	99.62	98.87	257
Acevedo	Vgg-16	94.33	99.61	96.84	95.31	99.61	96.76	1919
Khan	MLANET-FS-ELM	-	97.42	100.00	99.68	99.35	99.12	2487
Our Model III	CNN	97.84	98.15	99.85	94.31	100.00	98.03	3250

than five WBC types and used deep learning, hybrid, or image processing techniques. Rezatofighi et al. [31] applied image segmentation and LBP features with an SVM classifier on 251 LISC images. Acevedo et al. [2] used VGG-16 CNNs, also on the LISC dataset. Reena and Ameer [30] combined semantic segmentation with AlexNet for classification. Wang et al. [43] employed YOLOv3 for classifying 11 WBC categories. Khan et al. [17] used AlexNet-based CNNs with feature selection and an extreme learning machine, achieving 95.72˜97.79% accuracy depending on the FS method used (see: Fig. 3).

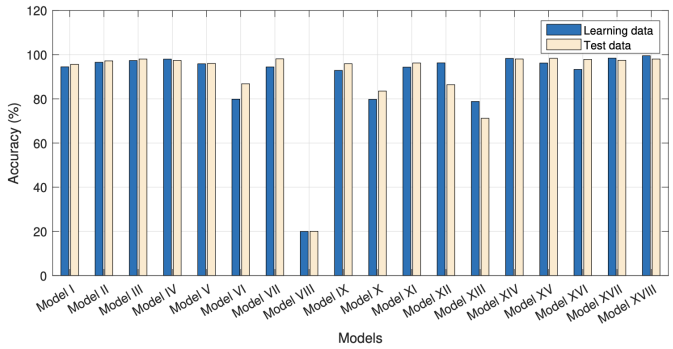


Fig. 3. Classification accuracy for learning and test data set across experiments.

A comparison of the classification accuracy of each of the five WBC classes considered, as well as the average accuracy of their classification, with the results obtained by other authors is shown in Table 6.

To highlight the robustness of our method, Table 6 includes the number of test images used in each study. Our test set is the largest, suggesting greater reliability due to its representativeness. The proposed approach achieves high classification accuracy across all WBC classes. Model III classifies Neutrophils with 100% accuracy, outperforming other methods. Its lowest result is for Monocytes (94.31%), though other approaches show similar or worse performance for this class. Basophils, eosinophils, and lymphocytes are classified with accuracy comparable to other methods. Overall, our method yields a higher mean clas-

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97554-7_29

sification accuracy than most others—surpassed only by the approaches in [30] and [17].

Khan et al. [17] achieved high accuracy but classified only four WBC types, excluding Basophils, simplifying the model. Recent studies show slightly better accuracy using advanced DL or hybrid networks.

The models presented in this study show high accuracy in automatic classification and counting of WBCs using CNNs. This research explored various network configurations, revealing the impact of layer depth, filter size, and optimization algorithms on classification accuracy.

Analysis results show that similar CNN performance can be achieved with different network configurations, yet finding the best one often requires numerous tests. The developed models demonstrate the potential of CNNs use in automated medical diagnostics, showing reasonable accuracy in automated blood cell analysis.

Future work may focus on expanding the dataset with more diverse and clinically varied samples to improve generalization. Incorporating explainable AI techniques could also enhance model transparency, making results more interpretable for medical professionals. Additionally, integrating the model into a real-time diagnostic support system would allow for practical validation in clinical settings.

References

1. Acevedo, A., et al.: Recognition of peripheral blood cell images using convolutional neural networks. Comp. Methods and Programs in Biomedicine **180**, 105020 (2019)
2. Acevedo, A., et al.: A new convolutional neural network predictive model for the automatic recognition of hypogranulated neutrophils in myelodysplastic syndromes. Computers in Biology and Medicine **134**, 104479 (2021)
3. Alférez, S., Merino, A., Acevedo, A., Puigví, L., Rodellar, J.: Color clustering segmentation framework for image analysis of malignant lymphoid cells in peripheral blood. Medical & biological engineering & computing **57**, 1265–1283 (2019)
4. Bailer, C., Habtegebrial, T., Stricker, D., et al.: Fast feature extraction with cnns with pooling layers. arXiv preprint arXiv:1805.03096 (2018)
5. Bain, B.J.: Diagnosis from the blood smear. New England Journal of Medicine **353**(5), 498–507 (2005)
6. CellaVision: CellaVision DM9600 (2023), https://www.cellavision.com/products/hardware/cellavisionr-dm9600, accessed: 2025-02-20
7. Cooper, C.: Blood: A very short introduction. Oxford University Press (2016)
8. Dong, N., et al.: A self-adaptive approach for white blood cell classification towards point-of-care testing. Applied Soft Computing **111**, 107709 (2021)
9. Dozat, T.: Incorporating Nesterov momentum into Adam (2016)
10. Drałus, G., Mazur, D., Czmil, A.: Automatic detection and counting of blood cells in smear images using retinanet. Entropy **23**(11), 1522 (2021)
11. Goodfellow, I., Bengio, Y., Courville, A., Bengio, Y.: Deep learning, vol. 1. MIT press Cambridge (2016)
12. Gupta, P.: What your blood tells? a review. Journal of Cell and Tissue Research **20**(2), 6897–6913 (2020)

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97554-7_29

13. Heaton, J.: Artificial Intelligence for Humans, Volume 3: Deep Learning and Neural Networks, vol. 3. Space Independent Publishing Platform (2015)
14. Hegde, R.B., et al.: Comparison of traditional image processing and deep learning approaches for classification of white blood cells in peripheral blood smear images. Biocybernetics and Biomedical Engineering **39**(2), 382–392 (2019)
15. Janocha, K., Czarnecki, W.M.: On loss functions for deep neural networks in classification. arXiv preprint arXiv:1702.05659 (2017)
16. Josephine, V.H., Nirmala, A., Alluri, V.L.: Impact of hidden dense layers in convolutional neural network to enhance performance of classification model. In: IOP Conference Series: Materials Science and Engineering. vol. 1131, p. 012007. IOP Publishing (2021)
17. Khan, A., Eker, A., Chefranov, A., Demirel, H.: White blood cell type identification using multi-layer convolutional features with an extreme-learning machine. Biomedical Signal Processing and Control **69**, 102932 (2021)
18. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
19. Kowal, M., Skobel, M., Gramacki, A., Korbicz, J.: Breast cancer nuclei segmentation and classification based on a deep learning approach. International Journal of Applied Mathematics and Computer Science **31**(1), 85–106 (2021)
20. Kowal, M., Żejmo, M., Korbicz, J.: Nuclei detection in cytological images using convolutional neural network and ellipse fitting algorithm. In: AI and Soft Computing: 17th Int. Conf., ICAISC 2018, Zakopane, Poland, June 3-7, 2018, Proceedings, Part II 17. pp. 157–167. Springer (2018)
21. Krizhevsky, A., et al.: Imagenet classification with deep convolutional neural networks. Advances in neural information processing systems **25** (2012)
22. Lin, T.Y., Dollár, P., Girshick, R., He, K., Hariharan, B., Belongie, S.: Feature pyramid networks for object detection. In: Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition. pp. 2117–2125 (2017)
23. McKinnon, K.M.: Flow cytometry: an overview. Current protocols in immunology **120**(1), 5–1 (2018)
24. Merino, A., Puigví, L., Boldú, L., Alférez, S., Rodellar, J.: Optimizing morphology through blood cell image analysis. International journal of laboratory hematology **40**, 54–61 (2018)
25. Nazlibilek, S., Karacor, D., Ercan, T., Sazli, M.H., Kalender, O., Ege, Y.: Automatic segmentation, counting, size determination and classification of white blood cells. Measurement **55**, 58–65 (2014)
26. Patan, K., Rutkowski, G.: Application of deep learning to seizure classification. In: Advances in Diagnostics of Processes and Systems: Selected Papers from the 14th Int. Conf. on Diagnostics of Processes and Systems (DPS), September 21–23, 2020, Zielona Góra (Poland). pp. 157–172. Springer (2020)
27. Patgiri, C., Ganguly, A.: Adaptive thresholding technique based classification of red blood cell and sickle cell using naïve bayes classifier and k-nearest neighbor classifier. Biomedical Signal Processing and Control **68**, 102745 (2021)
28. Putzu, L., Caocci, G., Di Ruberto, C.: Leucocyte classification for leukaemia detection using image processing techniques. AI in Medicine **62**(3), 179–191 (2014)
29. Qin, F., Gao, N., Peng, Y., Wu, Z., Shen, S., Grudtsin, A.: Fine-grained leukocyte classification with deep residual learning for microscopic images. Computer methods and programs in biomedicine **162**, 243–252 (2018)
30. Reena, M.R., Ameer, P.: Localization and recognition of leukocytes in peripheral blood: A deep learning approach. Computers in Biology and Medicine **126**, 104034 (2020)

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97554-7_29

31. Rezatofighi, S.H., Soltanian-Zadeh, H.: Automatic recognition of five types of white blood cells in peripheral blood. Computerized Medical Imaging and Graphics **35**(4), 333–343 (2011)
32. Rodellar, J., Alférez, S., Acevedo, A., Molina, A., Merino, A.: Image processing and machine learning in the morphological analysis of blood cells. International journal of laboratory hematology **40**, 46–53 (2018)
33. Safuan, S.N.M., Tomari, M.R.M., Zakaria, W.N.W.: White blood cell (WBC) counting analysis in blood smear images using various color segmentation methods. Measurement **116**, 543–555 (2018)
34. Shahin, A.I., Guo, Y., Amin, K.M., Sharawi, A.A.: White blood cells identification system based on convolutional deep neural learning networks. Computer methods and programs in biomedicine **168**, 69–80 (2019)
35. Skobel, M., Kowal, M., Korbicz, J., Obuchowicz, A.: Cell nuclei segmentation using marker-controlled watershed and bayesian object recognition. In: Information Technology in Biomedicine: Proceedings 6th Int. Conf., ITIB'2018, Kamień Śląski, Poland, June 18–20, 2018 6. pp. 407–418. Springer (2019)
36. Su, M.C., Cheng, C.Y., Wang, P.C.: A neural-network-based approach to white blood cell classification. The scientific world journal **2014**(1), 796371 (2014)
37. Szegedy, C., et al.: Going deeper with convolutions. In: Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition. pp. 1–9 (2015)
38. Vogado, L., et al.: Leukemia diagnosis in blood slides using transfer learning in cnns and svm for classification. Engineering Applications of AI **72**, 415–422 (2018)
39. Vogado, L., et al.: Diagnosis of leukaemia in blood slides based on a fine-tuned and highly generalisable deep learning model. Sensors **21**(9), 2989 (2021)
40. Wang, H., Adhikary, A.: Stressnet: A deep neural network based on dynamic dropout layers for stress recognition. In: Neural Information Processing: 28th Int. Conf. ICONIP, Bali, Indonesia, Dec.8–12, Part II 28. pp. 502–512. Springer (2021)
41. Wang, Q., Bi, S., Sun, M., Wang, Y., Wang, D., Yang, S.: Deep learning approach to peripheral leukocyte recognition. PloS one **14**(6), e0218808 (2019)
42. Wang, Y., Cao, Y.: Leukocyte nucleus segmentation method based on enhancing the saliency of saturation component. Journal of Algorithms & Computational Technology **13**, 1748302619845783 (2019)
43. Wang, Y., Cao, Y.: Quick leukocyte nucleus segmentation in leukocyte counting. Computational and mathematical methods in medicine **2019**(1), 3072498 (2019)
44. Ward, R., Wu, X., Bottou, L.: Adagrad stepsizes: Sharp convergence over nonconvex landscapes. Journal of Machine Learning Research **21**(219), 1–30 (2020)
45. West Medica: Hematology: Digital morphology of blood cells: Vision Hema (2019), https://wm-vision.com/en/product/hema, accessed: 2025-02-20
46. Whalan, J.E., Whalan, J.E.: Hematology glossary. A Toxicologist's Guide to Clinical Pathology in Animals: Hematology, Clinical Chemistry, Urinalysis pp. 109–143 (2015)
47. Zaheer, R., Shaziya, H.: A study of the optimization algorithms in deep learning. In: 2019 3rd Int. Conf. on inventive systems and control (ICISC). pp. 536–539. IEEE (2019)
48. Zheng, X., Wang, Y., Wang, G., Liu, J.: Fast and robust segmentation of white blood cell images by self-supervised learning. Micron **107**, 55–71 (2018)
49. Zou, F., Shen, L., Jie, Z., Zhang, W., Liu, W.: A sufficient condition for convergences of Adam and RMSProp. In: Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition. pp. 11127–11135 (2019)

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97554-7_29