electronics

MDPI

Article

Deep Learning Models for Classification of Red Blood Cells in Microscopy Images to Aid in Sickle Cell Anemia Diagnosis

Laith Alzubaidi [1,2,*], Mohammed A. Fadhel [2,3], Omran Al-Shamma [2], Jinglan Zhang [1] and Ye Duan [4]

1 Faculty of Science & Engineering, Queensland University of Technology, Brisbane, QLD 4000, Australia; jinglan.zhang@qut.edu.au
2 Al-Nidhal Campus, University of Information Technology & Communications, Baghdad 00964, Iraq; Mohammed.a.fadhel@uoitc.edu.iq (M.A.F.); o.al_shamma@uoitc.edu.iq (O.A.-S.)
3 College of Computer Science and Information Technology, University of Sumer, Thi Qar 64005, Iraq
4 Faculty of Electrical Engineering and Computer Science, University of Missouri, Columbia, MO 65211, USA; duanye@missouri.edu
* Correspondence: laith.alzubaidi@hdr.qut.edu.au; Tel.: +61-421933812

Received: 21 January 2020; Accepted: 1 March 2020; Published: 4 March 2020

Abstract: Sickle cell anemia, which is also called sickle cell disease (SCD), is a hematological disorder that causes occlusion in blood vessels, leading to hurtful episodes and even death. The key function of red blood cells (erythrocytes) is to supply all the parts of the human body with oxygen. Red blood cells (RBCs) form a crescent or sickle shape when sickle cell anemia affects them. This abnormal shape makes it difficult for sickle cells to move through the bloodstream, hence decreasing the oxygen flow. The precise classification of RBCs is the first step toward accurate diagnosis, which aids in evaluating the danger level of sickle cell anemia. The manual classification methods of erythrocytes require immense time, and it is possible that errors may be made throughout the classification stage. Traditional computer-aided techniques, which have been employed for erythrocyte classification, are based on handcrafted features techniques, and their performance relies on the selected features. They also are very sensitive to different sizes, colors, and complex shapes. However, microscopy images of erythrocytes are very complex in shape with different sizes. To this end, this research proposes lightweight deep learning models that classify the erythrocytes into three classes: circular (normal), elongated (sickle cells), and other blood content. These models are different in the number of layers and learnable filters. The available datasets of red blood cells with sickle cell disease are very small for training deep learning models. Therefore, addressing the lack of training data is the main aim of this paper. To tackle this issue and optimize the performance, the transfer learning technique is utilized. Transfer learning does not significantly affect performance on medical image tasks when the source domain is completely different from the target domain. In some cases, it can degrade the performance. Hence, we have applied the same domain transfer learning, unlike other methods that used the ImageNet dataset for transfer learning. To minimize the overfitting effect, we have utilized several data augmentation techniques. Our model obtained state-of-the-art performance and outperformed the latest methods by achieving an accuracy of 99.54% with our model and 99.98% with our model plus a multiclass SVM classifier on the erythrocytesIDB dataset and 98.87% on the collected dataset.

Keywords: red blood cells; erythrocytes; sickle cell anemia; deep learning; classification; transfer learning

1. Introduction

The blood molecule comprises various types of cells, of which red blood calls (RBCs) form a fundamental part. Meanwhile, anemia is a condition where there is a lack of healthy red blood cells to carry adequate oxygen in the blood, resulting in weariness [1]. Anemia is mainly categorized as acute or chronic. More specifically, the symptoms suddenly rise in acute anemia, while the symptoms gradually grow in chronic anemia. Different reasons cause anemia such as the production of abnormally shaped RBCs, gain in water weight throughout pregnancy, reduction in erythropoietin generation, inadequate intake of iron content, and blood loss throughout menstruation. In addition, a disorder produced due to the existence of RBCs with sickle shape is defined as drepanocytosis or sickle cell anemia (SCA). These types of cells are generated due to the transformation in the hemoglobin gene. If both parents have two abnormal genes, parents pass on genes to their child, and then the child will get the disease of SCA; hence, SCA is a hereditary disease. The variation in RBCs' shape is the main responsibility of hemoglobin S. This irregular shape variation reduces the oxygen delivery and, in turn, these cells will be stuck in the blood vessels. In addition, these cells tend to break into pieces [2,3]. Finally, based on the published data and statistics, Italy, Greece, Turkey, India, and South Africa are countries suffering the most from SCA [4]. In addition, there are around 100,000 Americans are suffering from it. Furthermore, it is the reason behind an extreme reduction in the rates of mortality among children who get SCA.

In recent years, several techniques for diagnosing SCA have been used. The manual technique for examining the blood smear has heavily relied on a pathologist's skills. It is a comprehensive and tiresome task. Furthermore, the excessive changeability of the edge, location, shape, and size makes the manual technique too complicated. On the other hand, the most commercially used tool for obtaining the blood cell count is called the hemocytometer. Since the device is expensive, it is not available in most hospitals. A computerized system helps the pathologists and physicians to recognize the correct type of anemia [5]. Several image processing and machine learning techniques have been utilized to classify, count, and segment sickle cell anemia in erythrocytes [6]. Although these techniques have shown good performance, they are sensitive to the different sizes, colors, and shapes of cells. Moreover, the good performance of these techniques relies on chosen features. These techniques take many steps (pre-processing, feature extraction, features selection, and classification) to implement the classification task. In recent years, deep learning has overcome these barriers and showed impressive performance in various tasks [7–9]. Deep learning has the ability to automatically extract features and implement the classification in one shot [10]. In this paper, we propose three deep learning models to classify erythrocytes in three classes, namely: circular (normal), elongated (sickle cells), and other blood content. One of the major issues of adopting deep learning models in red blood cell classification tasks is the lack of training data due to difficulty in collecting erythrocyte data and labeling. We have addressed this issue by offering different solutions including same domain transfer learning, data augmentation, and lightweight models. The proposed models accurately and effectively recognize the three types of erythrocytes which assist in evaluating the danger level of sickle cell anemia.

1.1. Challenges

The precise classification of sickle cell anemia in erythrocytes is a hard task in automatic medical diagnosis due to many challenges such as:

- Overlapping cells
- Lack of training data
- Low contrast between cells and background
- Heterogeneous and complex shapes and sizes of cells
- Required effective deep learning models for classification
- Various staining processes.

1.2. Contributions

The main contributions of this paper are listed below:

- We have designed three shallow convolutional neural network models to classify erythrocytes into three classes (circular [normal], elongated [sickle cells], and others).
- We have empirically applied the same domain transfer learning technique to address the lack of training data.
- We have trained proposed models with four different scenarios to explore the best training situation.
- Our results outperformed the results of the latest methods by achieving an accuracy of 99.54% on the erythrocytesIDB dataset and 98.87% on the collected dataset.
- We have trained a multiclass SVM classifier with extracted features by our model. It obtained an accuracy of 99.98% on the erythrocytesIDB dataset.
- We have introduced a concise review of the state-of-the-art methods in SCA classification and detection.

2. Literature Review

Several techniques for diagnosing SCA have been released, which can be categorized as image processing and machine learning methods. Patil et al. [11] introduced a technique for identifying and counting the RBCs. They used a threshold of global Otsu with a threshold value of 140 for converting the image to binary. Then, the labeling algorithm is employed for counting the cells. Alomari et al. utilized the Circular Hough Transform (CHT) algorithm with multi-iteration [12]. Bhagvathi and Thomas performed the red component extraction from the RGB image, while the FIS editor with rules is used for circulatory detecting [13]. Note that the RBCs count is the same as the number of circles. Furthermore, Hough transform is employed for separating the white blood cells (WBCs) away from the RBCs. Maitra and Gupta [14] set up a counter for counting the cells. To highlight the RBCs, the green component was extracted from the RGB image by Thejashwini and Padma [15]. Their cell count process was achieved by using CHT. Mazalan et al. [16] performed image cropping for obtaining small image partitions to determine the maximum and minimum radius. The RBCs count process was accomplished by using CHT. CHT used the computed radius values, which acted as input values to it. Tulsam et al. [17] applied the extraction of the second component of YCbCr format for highlighting the presence of RBCs in the blood smear. In addition, they set up a counter for counting the RBCs, while the watershed transformation technique is employed for separating the overlapped cells. They applied some morphological operations as well. Sreekumar and Bhattacharya [18] have utilized CHT for calculating the number of cells. In their approach, the WBCs have a more circular shape than that of the RBCs, as they assumed.

A distance tool is utilized for determining the radius range required by the CHT algorithm. In contrast, the mean radius value is calculated by Chintawar et al. [19] for detecting the sickle cells. Their system diagnosed 12 types of anemia disease. The proportion of each type is counted using a counter. Patil and Khot [20] computed the form factor for detecting the existence of various types of RBCs such as dacrocytes and elliptocytes. The classification method is based on a set of specific range values of the form factor. On the other hand, the fractal dimension approach is employed for identifying the sickle cells [21]. The radius mean, lowest, and highest values are calculated for analyzing the sickle cell shape where the standard cell size is employed as a scale. The sickle cells are identified based on the ratio (major/minor) axis length. Rexcy et al. [22] used the component Labeling algorithm for counting the cells, whereas Rakshita and Bhowmik [23] introduced an approach utilizing edge detection, as well as morphological operations for identifying the sickle cells. The edge detection is performed using a Sobel operator. The form factor was calculated as well. The abnormal cells have a form factor of less than 0.5, and those from 0.5 to 1 were considered normal cells. Revathi and Jeevitha [24] presented another approach for diagnosing the sickle cells. First, the gradient watershed transform is used for determining regional minima in real images. Second, the resulting gradient map is exposed

for detecting the nuclei. Third, the shape descriptors were employed for retaining the valid nuclei. Besides, other features were computed such as the mean square error, peak signal-to-noise ratio, and root mean square error. This approach achieved the well-behaved performance of ellipse adjustment utilizing natural images. Bala and Doegar [25] proposed watershed segmentation for detecting the sickle cells. The form factor was calculated for identifying the abnormal cells, while the contour plot was employed for highlighting the target and sickle cells. Hidalgo et al. [26] presented a technique for analyzing the shape of red blood cells smear of sickle cell diseases by utilizing ellipse adjustments and a new technique for identifying significant points. They also employed a set of constraints that aided in avoiding the use of image preprocessing steps. Parvathy et al. [27] presented watershed segmentation and Otsu's segmentation as an approach for detecting the sickle cells. Compactness and form factor were calculated for differentiating the abnormal cells from the normal ones. Veluchamy et al. [28] applied artificial neural networks for identifying normal/abnormal cells. Thresholding techniques and some morphological operations were used for segmenting the image. Next, the features were extracted using algebraic moment invariants and statistics of the first and second gray level. Lastly, the extracted features were analyzed for differentiating the normal/abnormal cells. Poomcokrak and Neatpisarnvani [29] employed neural networks as a technique for identifying the sickle cells. The individual RBCs were extracted using an edge detection algorithm. Then, the neural network was utilized for analyzing and classifying the individual cells. Fadhel et al. [30] introduced a watershed transform and the CHT for detecting the sickle cells. CHT was used for identifying the normal RBCs, which had a circular shape. While the effectiveness factor was calculated during the watershed segmentation for differentiating normal/sickle cells, the RBCs count was achieved using a counter. Sharma et al. [31] presented an approach for diagnosing thalassemia and SCA. The marker-controlled watershed transform was used for separating the touching erythrocytes. The extracted features were classified using the K Nearest Neighbor. Rodrigues et al. [32] introduced another approach for differentiating various types of erythrocytes. First, the image was converted to binary using the global Otsu's threshold. Next, feature extraction was performed after applying morphological operations.

Lastly, the features were categorized utilizing some classifiers, hence, selecting the best one. Chen et al. [33] presented a technique for diagnosing hemolytic anemia. The concavity information was employed for isolating the overlapped cells, whereas different classifiers were used for classifying different types of hemolytic anemia. Acharya et al. [34] presented an image processing method to split up the erythrocytes from other blood components and then classify them into 11 types using the K-Medoids algorithm. Elsalamony [35] has used the shape signature technique to detect two types of anemia then classify them using a neural network. Albayrak et al. [36] have implemented CHT to distinguish between healthy cells and sickle cells. Chy et al. [37] extracted features such as ratio, entropy, mean, standard deviation, and variance, which were used to train SVM to classify normal and sickle cells. Chy et al. [38] have employed Extreme Learning Machine (ELM), Support Vector Machine (SVM), and K Nearest Neighbor (KNN) classifiers to classify normal verse sickle cells. Recently, Convolutional Neural Networks (CNNs) have been used to classify RBC and diagnose sickle cell disease [39,40]. These methods achieved accuracies of 86.34% and 87.50% respectively. These accuracies are still not the goal for the sickle cell disease classification task, since the precise classification is the first step toward accurate diagnosis.

We have summarized the research problems of the image processing methods that are mentioned above:

- Not paying attention to WBCs in an image, which could lead to a false diagnosis
- Not automated
- Using a few samples for testing
- Failing to handle the overlapped cells
- Not accurate.

We have also summarized the research problems of the machine learning methods that are mentioned above:

- Ignore other blood components, which may lead to the wrong classification
- Binary classification (normal cells versus sickle cells)
- Sensitive to different sizes, colors, and complex shapes
- Require domain-specific expertise on the feature extraction and a limited number of features
- Time-consuming.

Research Gap: The precise classification of RBCs is important to have a perfect cell count. As a result, it will give a clear image of the danger level of sickle cell anemia. Image processing and machine learning methods for sickle cell anemia classification tasks fell in several issues as mentioned above. Therefore, it failed to achieve an accurate classification performance. On the other hand, CNN has been employed as a sickle cell anemia classification task to overcome the barriers of image processing and machine learning methods. However, CNN methods achieved a low accuracy of classification due to small datasets of RBCs being used for training. CNN models require a large amount of data to perform well. Hence, it is essential to deal with issue regarding the lack of data. This is one of the reasons that the CNN methods, which worked on the sickle cell anemia classification task dataset, accomplished low accuracies. In this paper, we address the issue of the lack of training data for CNN models.

3. Methodology

The methodology is composed of five sections: Datasets, Lack of Training Data, the Architecture of Proposed Models, Training Scenarios, and Feature Extraction Using Proposed Models.

3.1. Datasets

We have utilized three datasets in this paper. Dataset 1, which is a public dataset, is the main target of this paper. Dataset 2 is only employed for transfer learning purposes. Dataset 3 is a collected dataset to prove the robustness of the proposed models.

3.1.1. Dataset 1 (Target Dataset)

The erythrocytesIDB dataset has images of peripheral blood smears samples taken from patients with Sickle Cell Disease in the Special Hematology Department of the General Hospital from Santiago de Cuba [26]. It has images of three tasks (segmentation, detection, classification), and we work on the first task, which is the classification (erythrocytesIDB1). The dataset has 196 full-field images with 626 images of individual cells classified as circular (202), elongated (211), or other (213). Each cell has a size of 80 × 80 pixels. Figure 1 shows the samples of individual cells of the dataset.

3.1.2. Dataset2

This dataset has been utilized in this paper for transfer learning purposes. It has been collected from three sub-datasets. The first one has 367 images of white blood cells as well as other blood components [41]. This dataset is utilized to classify four types of white blood cells (neutrophils, eosinophils, lymphocytes, and monocytes). Each image has a size of 640 × 480 pixels. The second one has 150 blood images with a size of 400 × 298 from Wadsworth center [42,43]. The third one is called ALL-IDB2, which consists of 260 images of lymphocytes utilized for the diagnosis of acute lymphoblastic leukemia. The images are classified into two classes: normal lymphocytes (130 images) and immature lymphocytes (130 images) [43,44]. Each image has a size of 257 × 257. We have divided each image into small patches of 80 × 80 pixels to fit within the input size of the proposed models and the total number of patches is 22,206. Figure 1 shows the samples of individual cells of the dataset.

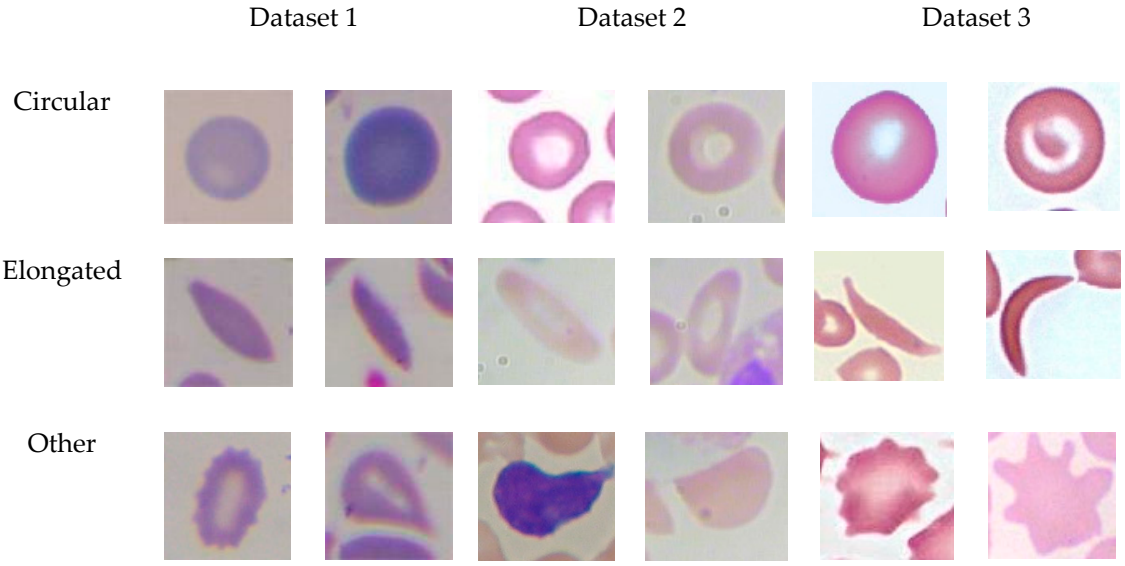

Figure 1. Samples of the individual cells of datasets 1–3.

3.1.3. Dataset3

We have collected 200 red blood smears samples from different websites [45] and an Internet search. We have extracted a total of 726 individual patches in the size of 80 × 80 pixels. Extracted patches have been classified by an expert in the field into three categories: circular cells or normal erythrocytes (292), elongated cells, which are sickle cell anemia (201) or other cells (233). Figure 1 shows samples of individual cells of the dataset.

3.2. Lack of Training Data Issue

Deep learning is utilized for classifying several types of medical images and showed great performance [10,46,47]. However, some limitations are frequently encountered when utilizing deep learning. One of the major limitations is the lack of training data, which is presented in this section along with solutions.

Deep learning is extremely data-hungry, as it also has representation learning [48]. Deep learning demands an extensive amount of data to achieve a well-behaved performance model. As the data increases, an extremely well-behaved performance can be achieved. In most conditions, the available data are sufficient to get a good performance model, but sometimes, there is a lack of data for directly using deep learning [49]. In handling this problem, two solutions are suggested in this paper to address the issue of the lack of training data surrounding the sickle cell classification task. The first solution is employing a transfer learning technique. Transfer learning is a technique where deep learning models are trained on a large dataset, and then the models are fine-tuned to train on a small target dataset, as shown in Figure 2.

Transfer learning is very important due to the limited datasets in the medical and healthcare domain; otherwise, the convergence of weights will not be possible. It is generally used to exploit the deep learning power in handling biomedical data [50] since it has its own field, and it is a significant machine learning trend [51].

Many researchers have utilized pre-trained models (such as AlexNet, GoogleNet, and ResNet), which trained on an ImageNet dataset that has 1000 classes of nature images such as animals and cars and then fine-tuned the models for medical tasks [52]. Although transfer learning from the ImageNet dataset improved the performance in many computer vision and pattern recognition tasks [53,54], the medical task is a quite different domain, and transfer learning could not improve the performance significantly [55]. It has been proven that transfer learning does not significantly affect performance on medical imaging tasks, with lightweight models trained from scratch performing nearly as well

as standard ImageNet transferred models [55]. The reason behind that is models that trained on the ImageNet dataset learned features that are completely different from medical image features. Furthermore, pre-trained models such as GoogleNet and ResNet are very deep, which requires a huge amount of training data to perform well, even with using transfer learning. However, the available data of red blood cells is not sufficient to train these models. Another issue is that the patch size of red blood cells is very small (input size: 80×80) to fit within the input size of the pre-trained models (input size: 224×224). Resizing the red blood cell patches leads to deteriorating their quality. As a result, that will affect the classification performance.

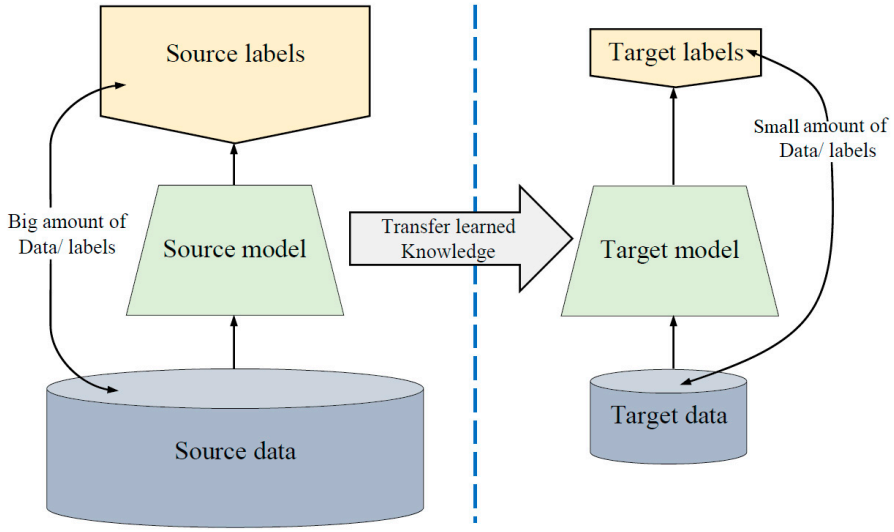

Figure 2. Transfer learning diagram.

To solve that, employing transfer learning from the same domain of the target dataset is the solution that we propose in this research to address the lack of training data of sickle cell disease classification tasks. This solution could significantly optimize performance. We have collected images that are similar to the target task images for the purpose of the transfer-learning technique. These collected images will not directly enlarge the actual images of the target task. Instead, these images can enhance the learning stage for a well-behaved representation of the initial layers of the model as well as a well-behaved mapping function and boost the model performance [50–56].

The steps of transfer learning that we implement are:

1. Training our proposed models (explained in the next section) on dataset2, which has images that are in the same domain of the target dataset.
2. Loading the pre-trained models. The first layers learned low-level features such as colors and edges, while last layers learned task-specific features.
3. Replacing the final layers with new layers to learn features specific to the target task.
4. Training the fine-tuned models with the target dataset, which is Dataset 1.
5. Predicting and evaluating the model accuracy.
6. Deploying the results.

The second solution is performing data augmentation [57]. Data augmentation is a process to increase the amount of training data to enhance classification performance. The augmentation process is achieved by applying several transformations. In this paper, we have applied rotation with seven different angles (45, 90, 125, 180, 225, 280, and 325) as shown in Figure 3. We also applied two flipping techniques (horizontal, vertical) and two different brightness degrees. The original images have been duplicated 11 times. Data augmentation and transfer learning aid to prevent the overfitting issue, since they both offer a large amount of data. The overfitting issue will occur if the CNN model has more

learning capacity than the dataset has information. However, our proposed models are shallow, which aids in avoiding the overfitting issue.

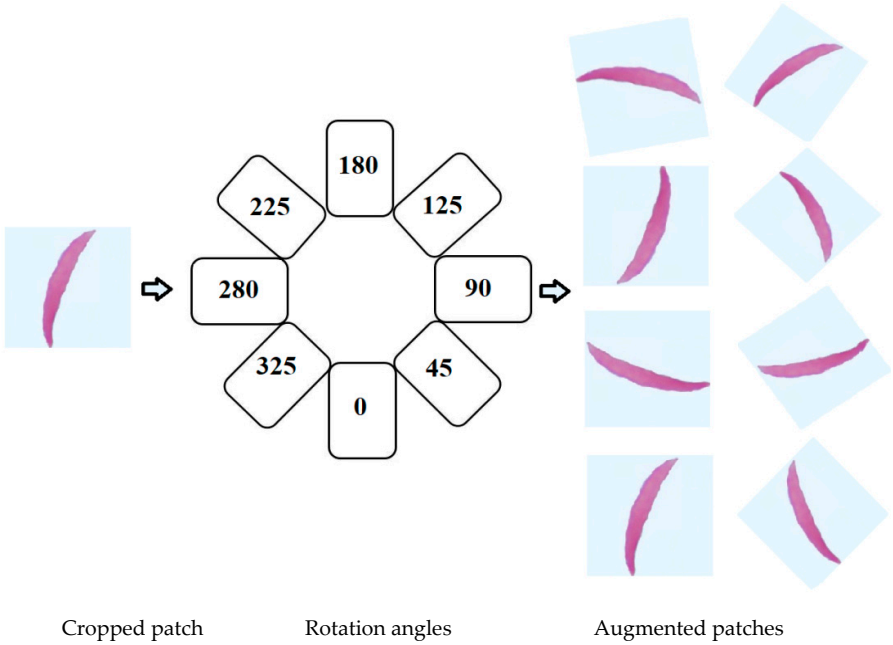

Figure 3. Rotation process.

3.3. The Architecture of Proposed Models

The latest designs of network architecture [58–60] show that the network classification performance is incredibly enhanced by redesigning the CNN structure in a way that facilitates the deep feature learning. Stacking more layers to a traditional Convolutional Neural Network such as VGG [61] and AlexNet [62] to make it very deep does not lead to better performance; instead, it leads to worse performance due to the gradient vanishing issue [63]. Hence, it is not always a solution by just increasing the depth by itself. Furthermore, very deep networks require a huge data amount to train well; therefore, we have designed lightweight models with multiple convolutions working in parallel. Redesigning the CNN structure in a way where several branches are fused (either concatenation or summation) is very beneficial for gradient propagation, as the error can be backpropagated through multiple paths. It also aids in combining different levels of features at each step of the network [60].

In this paper, we propose three lightweight models that are different regarding the number of layers and filters. The purpose of designing three models is to obtain the optimizable one for the erythrocyte classification. These models aggregated two modes of convolution neural networks which are traditional and parallel convolutional layers. The traditional convolutional layers are employed at the beginning of the models to reduce the size of the input image. After the traditional convolutional layers, parallel convolutional layers have been employed with different filter sizes to have different feature representations.

3.3.1. Model 1 Architecture

This model consists of 40 layers in total with nine convolutional layers. Each convolutional layer in the model is followed by batch normalization and Rectified Linear Unit (ReLU) to speed up the training process and to prevent vanishing gradient problem. At the beginning of the model, three traditional convolutional layers are used with filter sizes (1×1, 3×3, 5×5) to reduce the size of the input image. The output of traditional layers feeds to the first block of parallel convolutional layers. This block consists of three convolutional layers with filter sizes (1×1, 3×3, 5×5). These parallel convolutional layers are working together; then, the outputs concatenate in the concatenation

layer, which is followed by the batch normalization layer. The output of the first block passes to the second block of parallel convolutional layers, which is the same structure of the first block. After that, the output of the second concatenation layer passes to the average pooling layer to reduce the dimensionality. Three fully connected layers have been employed, and two dropout layers are between these fully connected layers to avoid overfitting [64]. Lastly, the softmax function is utilized to yield the classification output. All the details of the architecture of Model 1 are explained in Figure 4 (model 1) and Table 1.

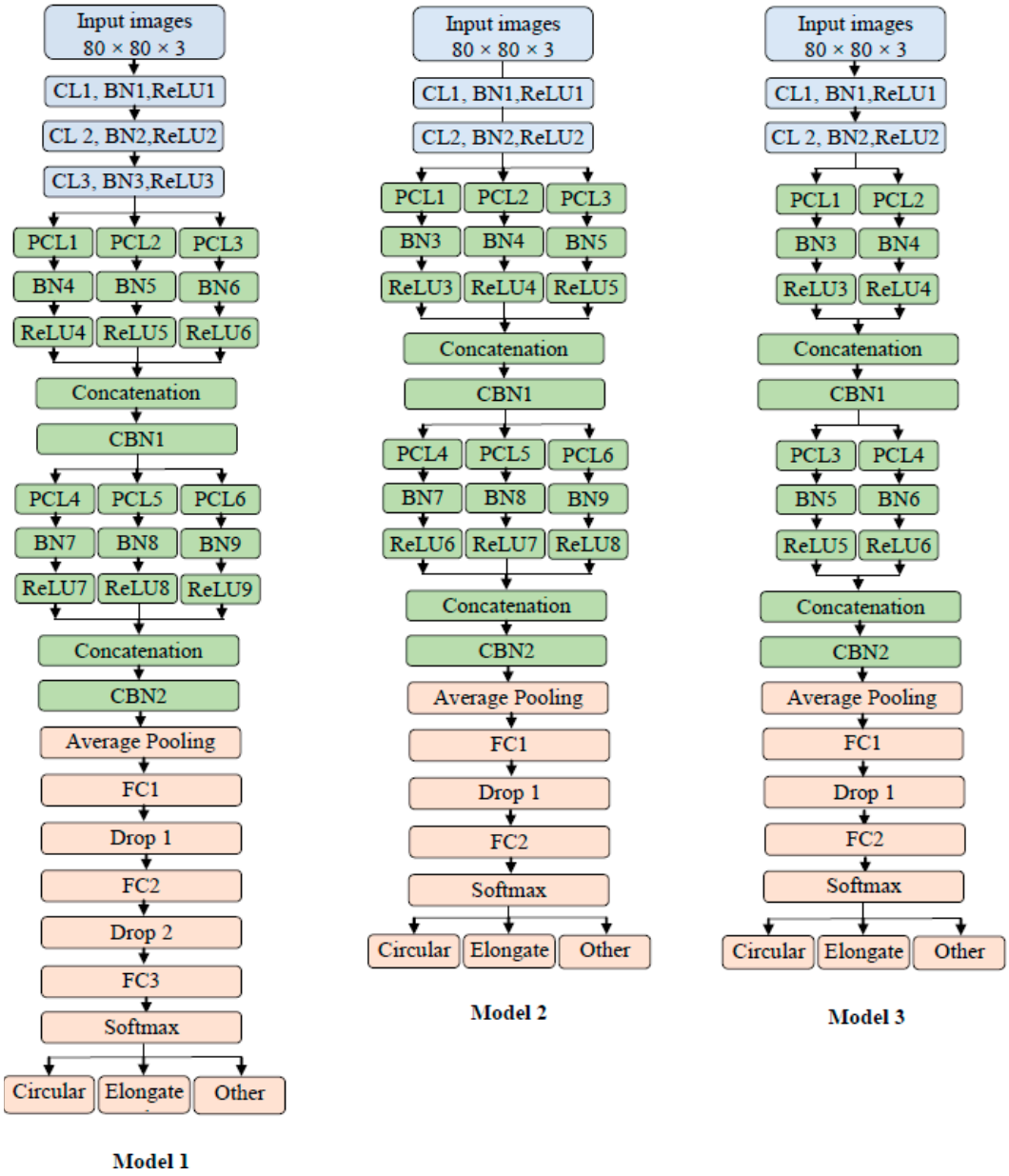

Figure 4. Proposed models' architecture.

3.3.2. Model 2 Architecture

This model consists of 35 layers in total with eight convolutional layers. As explained in Figure 4 (model 2) and Table 2, Model 2 has almost the same structure of Model 1 except that they are different in the following points:

- Different filters number (the depth)
- Model 2 has two traditional convolutions instead of the three in Model 1

- Model 2 has two fully connected layers and one dropout layer
- The average pooling layer has a filter size of 7×7 in Model 2, while in Model 1, the average pooling has a filter size of 4×4.

Table 1. Proposed Model 1 architecture, in which CL represents the Convolutional layer, PCL represents the Parallel Convolutional layer, BN represents the Batch Normalization layer, ReLU represents the Rectified Linear Unit, the drop represents the Dropout layer, and FC represents a fully connected layer.

Name of Layer	Filter Size (FS) and Stride (S)	Activations
Input layer	-	$80 \times 80 \times 3$
CL 1, BN 1, ReLU 1	FS = 1×1, S = 1	$80 \times 80 \times 64$
CL 2, BN 2, ReLU 2	FS = 3×3, S = 2	$40 \times 40 \times 64$
CL 3, BN 3, ReLU 3	FS = 5×5, S = 2	$20 \times 20 \times 64$
PCL 1, BN 4, ReLU 4	FS = 1×1, S = 1	$20 \times 20 \times 128$
PCL 2, BN 5, ReLU 5	FS = 3×3, S = 1	$20 \times 20 \times 128$
PCL 3, BN 6, ReLU 6	FS = 5×5, S = 1	$20 \times 20 \times 128$
Concatenation Layer 1	Three inputs	$20 \times 20 \times 384$
CBN 1	Batch Normalization	$20 \times 20 \times 384$
PCL 4, BN 7, ReLU 7	FS = 1×1, S = 2	$10 \times 10 \times 256$
PCL 5, BN 8, ReLU 8	FS = 3×3, S = 2	$10 \times 10 \times 256$
PCL 6, BN 9, ReLU 9	FS = 5×5, S = 2	$10 \times 10 \times 256$
Concatenation Layer 2	Three inputs	$10 \times 10 \times 768$
CBN 2	Batch Normalization	$10 \times 10 \times 768$
Average Pooling	Size = 4×4, S = 2	$4 \times 4 \times 768$
FC1	600	$1 \times 1 \times 600$
Drop 1	Learning rate: 0.5	$1 \times 1 \times 600$
FC 2	150	$1 \times 1 \times 150$
Drop 2	Learning rate: 0.5	$1 \times 1 \times 150$
FC 3	3	$1 \times 1 \times 3$
Softmax and Classoutput	3 classes: circular, elongated, other	-

Table 2. Proposed Model 2 architecture, in which CL represents the Convolutional layer, PCL represents the Parallel Convolutional layer, BN represents the Batch Normalization layer, ReLU represents the Rectified Linear Unit, the drop represents Dropout layer, and FC represents a fully connected layer.

Name of Layer	Filter Size (FS) and Stride (S)	Activations
Input layer	-	$80 \times 80 \times 3$
CL 1, BN 1, ReLU 1	FS = 3×3, S = 1	$80 \times 80 \times 16$
CL 2, BN 2, ReLU 2	FS = 5×5, S = 2	$40 \times 40 \times 16$
PCL 1, BN 3, ReLU 3	FS = 1×1, S = 1	$40 \times 40 \times 32$
PCL 2, BN 4, ReLU 4	FS = 3×3, S = 1	$40 \times 40 \times 32$
PCL 3, BN 5, ReLU 5	FS = 5×5, S = 1	$40 \times 40 \times 32$
Concatenation Layer 1	Three inputs	$40 \times 40 \times 96$
CBN 1	Batch Normalization	$40 \times 40 \times 96$
PCL 4, BN 6, ReLU 6	FS = 1×1, S = 2	$20 \times 20 \times 64$
PCL 5, BN 7, ReLU 7	FS = 3×3, S = 2	$20 \times 20 \times 64$
PCL 6, BN 8, ReLU 8	FS = 5×5, S = 2	$20 \times 20 \times 64$
Concatenation Layer 2	Three inputs	$20 \times 20 \times 192$
CBN 2	Batch Normalization	$20 \times 20 \times 192$
Average Pooling	Size = 7×7, S = 2	$7 \times 7 \times 192$
FC 1	90	$1 \times 1 \times 90$
Drop 1	learning rate:0.5	$1 \times 1 \times 90$
FC 2	3	$1 \times 1 \times 3$
Softmax and Classoutput	3 classes: circular, elongated, other	-

3.3.3. Model 3 Architecture

This model consists of 29 layers in total with six convolutional layers, as explained in Figure 4 (Model 3) and Table 3. By comparing Model 2 and model 3, we can find that Model 3 has almost the same structure as that of Model 2 except that they are different in the following points:

- Different filter number (the depth)
- Model 3 has two parallel convolutions in both blocks instead of three as in Model 2.

Table 3. Proposed Model 3 architecture, in which CL represents the Convolutional layer, PCL represents the Parallel Convolutional layer, BN represents the Batch Normalization layer, ReLU represents the Rectified Linear Unit, the drop represents the Dropout layer, and FC represents a fully connected layer.

Name of Layer	Filter Size (FS) and Stride (S)	Activations
Input layer	-	$80 \times 80 \times 3$
CL 1, BN 1, ReLU 1	FS = 3×3, S = 1	$80 \times 80 \times 16$
CL 2, BN 2, ReLU 2	FS = 5×5, S = 2	$40 \times 40 \times 16$
PCL 1, BN 3, ReLU 3	FS = 1×1, S = 1	$40 \times 40 \times 16$
PCL 2, BN 4, ReLU 4	FS = 3×3, S = 1	$40 \times 40 \times 16$
Concatenation Layer 1	Three inputs	$40 \times 40 \times 32$
CBN 1	Batch Normalization	$40 \times 40 \times 32$
PCL 3, BN 5, ReLU 5	FS = 1×1, S = 2	$20 \times 20 \times 32$
PCL 4, BN 6, ReLU 6	FS = 3×3, S = 2	$20 \times 20 \times 32$
Concatenation Layer 2	Three inputs	$20 \times 20 \times 64$
CBN 2	Batch Normalization	$20 \times 20 \times 64$
Average Pooling	Size = 7×7, S = 2	$7 \times 7 \times 64$
FC 1	60	$1 \times 1 \times 60$
Drop 1	Learning rate: 0.5	$1 \times 1 \times 60$
FC 2	3	$1 \times 1 \times 3$
Softmax and Classoutput	3 classes: circular, elongated, other	$1 \times 1 \times 3$

3.4. Training Scenarios

We have trained the three proposed models with several training scenarios to achieve better performance and deal with the lack of training data. We have utilized all four scenarios (described below) with Dataset 1 and then all four scenarios with Dataset 3. As mentioned above, Dataset 2 has been only used for transfer learning purposes to make the proposed models pre-trained models.

1. Scenario 1: Training the models on the original images of the datasets.
2. Scenario 2: Training the models on the original images of the datasets plus augmented images.
3. Scenario 3: First, we trained the models on Dataset 2 plus augmented images of Dataset 2 for transfer learning purposes. Then, the models were fine-tuned to train on the original images of the datasets.
4. Scenario 4: We repeated Scenario 3 but added in the augmented images with original images.

The three models with all scenarios have been trained for 80 epochs using Matlab 2019a as software and Graphics Processing Unit (GPU) as hardware. Some of the learnable filters from Model 2 trained with Scenario 4 are shown in Figure 5. This figure shows how learning improved from the first layer and the second layer. It is worth mentioning the training time of our models was short due to the lightweight architecture of our models and using GPUs.

3.5. Feature Extraction Using Proposed Models

We have utilized the extracted features by the proposed models to train a multiclass SVM classifier. The main aim of implementing this step is to prove that our models are robust and effective in the features extraction step. The process steps are:

1. Loading the data that proposed models tested on for the test phase.

2. Loading the trained proposed models to the workspace.
3. Extracting the learned features and labels from the last fully connected layer, which represents the last layer before the classifier. The models build a hierarchical representation of input data. Deeper layers hold higher-level features, which were built utilizing the lower-level features of earlier layers. Therefore, we extracted the features from the last fully connected layer.
4. Using the extracted features to train a multiclass support vector machine (SVM) classifier. This classifier has been utilized from the Statistics and Machine Learning Toolbox in MATLAB 2019a.
5. Classifying the test images using the trained SVM.
6. Calculating the accuracy on the test images.
7. Plotting some samples of test images with their predicted labels (Figure 6).

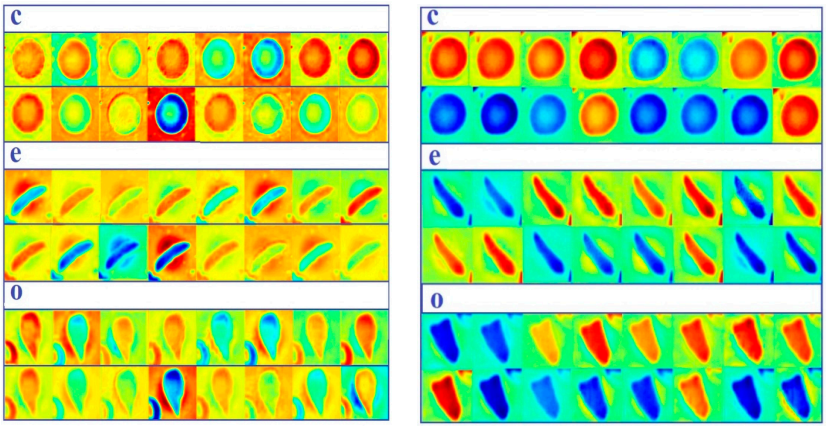

Figure 5. Some of the learnable filters from Model 2 trained with Scenario 4. c represents circular, e represents elongated, and o represents other. The first column is learnable filters from the first convolutional layer; the second column is learnable filters from the second convolutional layer.

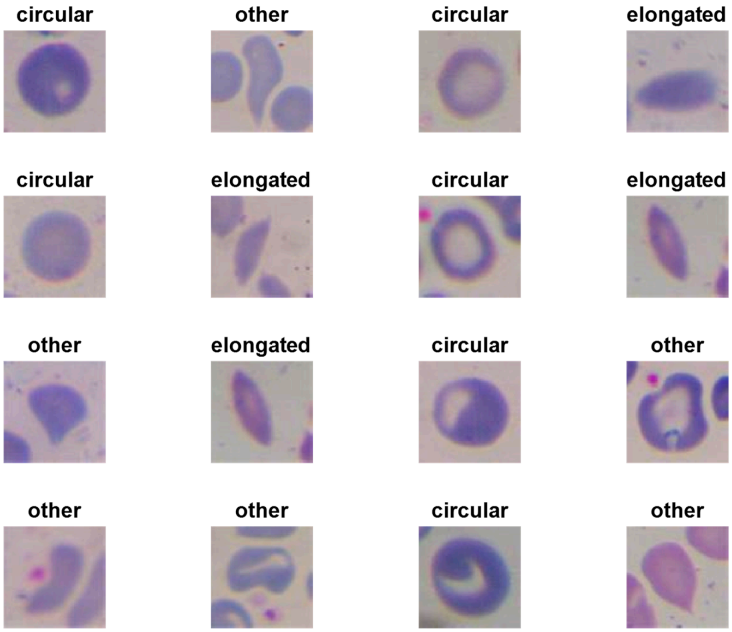

Figure 6. Some samples of test images with their predicted labels tested on Model 2 with the multiple support vector machine (SVM) classifier.

4. Experimental Results

We have evaluated our model in terms of accuracy, as written in eq1. The accuracy is the ratio between the true classifications and the total number of images. We have divided the data into 80% for the training phase and 20% for the testing phase.

$$\text{Accuracy} = (\text{TP} + \text{TN}) / (\text{TP} + \text{TN} + \text{FP} + \text{FN}) \quad (1)$$

As reported in Table 4, in the case Dataset 1(erythrocytesIDB1) with all four scenarios, Model 2 has achieved the highest accuracies in all first four scenarios by scoring 88.60% in Scenario 1, 93.87% in Scenario 2, 98.40% in Scenario 3, and 99.54% in Scenario 4. Model 1 has achieved the lowest accuracies in Scenario 1, Scenario 2, and Scenario 3 by scoring 85.40%, 91.21%, and 97.43%, respectively. Furthermore, it came in the second position in Scenario 4 by scoring 98.57%. Lastly, Model 3 has achieved the second-highest results in Scenario 1, Scenario 2, and Scenario 3 by scoring 86.17%, 92.34%, and 98.06%, respectively, and it scored the lowest accuracy in Scenario 4 by achieving 98.31%.

Table 4. Accuracy results of our scenarios with the proposed models.

	Scenarios	Model 1 (%)	Model 2 (%)	Model 3 (%)
Dataset 1 (erythrocytesIDB1)	Scenario 1	85.40	88.60	86.17
	Scenario 2	91.21	93.87	92.34
	Scenario 3	97.43	98.40	98.06
	Scenario 4	98.57	99.54	98.31
Dataset 3	Scenario 1	82.63	83.99	83.50
	Scenario 2	90.11	92.01	91.12
	Scenario 3	95.65	97.17	96.55
	Scenario 4	97.66	98.87	97.61

In the case of Dataset 3 with all four scenarios, Model 2 has also achieved the highest accuracies in all four scenarios by scoring 83.99% in Scenario 1, 92.01% in Scenario 2, 97.17% in Scenario 3, and 98.87% in Scenario 4. Again, Model 1 has achieved the lowest accuracies in Scenario 1, Scenario 2, and Scenario 3 by scoring 82.63%, 90.11%, and 95.65%, respectively. It also scored the second-highest accuracy in Scenario 4 by scoring 97.66%. Lastly, Model 3 has achieved the second-highest results in Scenario 1, Scenario 2, and Scenario 3 by scoring 83.50%, 91.12%, and 96.55%, respectively, and the lowest accuracy in Scenario 4 by achieving 97.61%.

From the performance of proposed models on Dataset 1 and Dataset 3, we can conclude the following points:

- Image augmentation and transfer learning techniques have significantly improved the classification performance of red blood cells classification. Transfer learning from the same domain offers a large benefit to performance.
- Due to using transfer learning from white blood images, our models handled the problem of not paying attention to WBCs in an image, which could lead to a false diagnosis.
- Proposed models are much smaller and shallower than the standard ImageNet models; they performed effectively on two different datasets and handled different cases of overlapped cells.
- Proposed models consumed less training data than other state-of-the-art CNN models such as GoogleNet and ResNet due to the shallow structure of models.
- Overall, all models have achieved high accuracies with Scenarios 3 and 4 compared to prior methods.
- Model 2 has the average number of layers and filters between Model 1 and Model 3, which assisted the model in having sufficient features to differentiate between classes. Model 1 has the largest number of layers and filters compared to Model 2 and Model 3, which requires more big training data; therefore, it achieved the lowest accuracies with Scenarios 1, 2, and 3 of both datasets (1, 3) where there is less training data. It achieved an accuracy that is better than Model 3 when the

training set is as big as that in Scenario 4 for both datasets (1, 3). Model 3 has the least number of layers and filters compared to Model 1 and Model 2, which requires a smaller amount of training data; therefore, It achieved higher accuracies with Scenarios 1, 2, and 3 of both datasets (1, 3) than Model 3. While it achieved an accuracy that is less than that of Model 3 when the training set is big as in Scenario 4 of both datasets (1, 3), it requires more layers and filters to differentiate between classes.

- Generally, all proposed models are lightweight models that aided in having a very fast training process besides using GPUs.
- Last but not least, we designed Model 3 first; then, we evaluated its performance. To improve the performance, we increased the width of the model as in Model 2, and it showed good improvement. To improve the performance further, we increased the depth of the network as in Model 1, but that did not improve the performance. Therefore, Model 2 has the best performance.

By employing the multiclass SVM classifier, the results were improved in some places, as reported in Table 5. Model 2 achieved the highest accuracies compared to Model 1 and Model 3 in four scenarios and the highest accuracy among the four scenarios is 99.98% with Scenario 4. Generally, the results of the three models in Scenario 4 are convergent and higher than other scenarios due to a large number of features that allow SVM to perform effectively. It is obvious that the accuracy of all models with Scenario 1 improved with SVM, yet it still has the lowest accuracies compared to the results of other scenarios. The results of the three models in Scenarios 2 and 3 are higher than those of Scenario 1 and less than those of Scenario 4.

Table 5. Accuracy results of our scenarios with proposed models and SVM classifier on the erythrocytesIDB1dataset.

Scenarios	Model 1 (%)	Model 2 (%)	Model 3 (%)
Scenario1	87.10	90.80	89.88
Scenario2	92.27	94.87	92.79
Scenario3	97.09	98.10	97.16
Scenario4	99.57	99.98	99.44

As cited in Table 6, Model 2 has outperformed the latest methods. It achieved state-of-the-art performance in terms of the classification of red blood cells. It is worth mentioning that all the model results in Scenarios 3 and 4 with Dataset 1 outperformed these methods. It is also worth mentioning that the testing time of our models was very short due to the lightweight architecture of our models and using GPUs. From that, we can conclude that our models are efficient. Lastly, some of the methods in Table 6 used hand-crafted features to train machine learning classifiers such as SVM, Naive Bayes, and KNN. The performance of these methods is not as good as our models with SVM. This proves our models extracted excellent features to train SVM. The performance of these classifiers relies on extracted features.

Table 6. The latest and highest accuracies of previous methods on Dataset 1 (erythrocytesIDB1) compared to our method.

Method	Accuracy (%)
Gual-Arnau et al. 2015 [65]	96.10
Rodrigues et al. 2016 [32]	93.18
Rodrigues et al. 2016 [32]	93.07
Rodrigues et al. 2016 [32]	94.59
de Faria et al. 2018 [66]	92.52
de Faria et al. 2018 [66]	93.67
Our method (Scenario 4, Model 2)	99.54
Our method (Scenario 4, Model2 +SVM)	99.98

5. Conclusions

We have proposed three deep learning models to classify three classes of red blood cells (erythrocytes) which are circular (normal), elongated (sickle cells), and other blood content. Two modes of convolution neural networks, which are traditional and parallel convolutional layers, have been utilized to design the proposed models. These models are dissimilar in the number of layers and learnable filters. We have empirically found the best model among all three models. The issue of the lack of training data of deep learning models in the red blood cell classification task was addressed by utilizing two techniques, which in this paper are transfer learning and data augmentation. The same domain transfer learning was utilized by collecting images from the same domain of the target dataset and training proposed models to be pre-trained models. Then, these pre-trained models were fine-tuned for the classification task of sickle cell disease. Two datasets have been used to evaluate our models. The experiment results proved that using the same domain transfer learning has significantly improved the performance of the sickle cell disease classification task. It also assisted our models with outperforming the latest erythrocyte classification methods. Our models have robustly classified erythrocytes with a high degree of accuracy. Finally, we proved that our models are effective in extraction features by training a multiclass SVM classifier, which showed excellent results. As future work, we plan to employ our pre-trained model for a white blood disease classification task.

Author Contributions: Conceptualization, L.A., M.A.F., and J.Z.; methodology, L.A., M.A.F., and J.Z.; software, L.A., J.Z.; validation, J.Z., Y.D., and O.A.-S.; data curation, L.A., M.A.F., and O.A.-S.; writing—original draft preparation, L.A., and O.A.-S.; writing—review and editing, L.A., M.A.F., O.A.-S., J.Z., Y.D.; supervision, J.Z, Y.D.; project administration, J.Z., Y.D. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the Queensland University of Technology.

Acknowledgments: The authors would like to thank all the reviewers for their valuable comments that improved our work. Moreover, special thanks to the Queensland University of Technology for supporting us.

Conflicts of Interest: The authors declare that they have no conflict of interest.

References

1. Stuart, M.J.; Nagel, R.L. Sickle-cell disease. *Lancet* **2004**, *364*, 1343–1360. [CrossRef]
2. Wąsowicz, M.; Grochowski, M.; Kulka, M.; Mikołajczyk, A.; Ficek, M.; Karpieńko, K.; Cićkiewicz, M. Computed aided system for separation and classification of the abnormal erythrocytes in human blood. In Biophotonics—Riga 2017. *Int. Soc. Opt. Photonics* **2017**, *10592*, 105920A.
3. Alzubaidi, L.; Fadhel, M.A.; Al-Shamma, O.; Zhang, J. Robust and Efficient Approach to Diagnose Sickle Cell Anemia in Blood. In *International Conference on Intelligent Systems Design and Applications*; Springer: Cham, Switzerland, December 2018; pp. 560–570.
4. Survey Data. Available online: https://www.cdc.gov/ncbddd/sicklecell/data.html (accessed on 25 December 2019).
5. Acharya, V.; Prakasha, K. Computer-Aided Technique to Separate the Red Blood Cells, Categorize them and Diagnose Sickle Cell Anemia. *J. Eng. Sci. Technol. Rev.* **2019**, *12*, 2. [CrossRef]
6. Das, P.K.; Meher, S.; Panda, R.; Abraham, A. A Review of Automated Methods for the Detection of Sickle Cell Disease. *IEEE Rev. Biomed. Eng.* **2019**, *13*, 309–324. [CrossRef] [PubMed]
7. Huang, Z.; Lin, J.; Xu, L.; Wang, H.; Bai, T.; Pang, Y.; Meen, T.-H. Fusion High-Resolution Network for Diagnosing ChestX-ray Images. *Electronics* **2020**, *9*, 190. [CrossRef]
8. Nurmaini, S.; Darmawahyuni, A.; Sakti Mukti, A.N.; Rachmatullah, M.N.; Firdaus, F.; Tutuko, B. Deep Learning-Based Stacked Denoising and Autoencoder for ECG Heartbeat Classification. *Electronics* **2020**, *9*, 135. [CrossRef]
9. Alzubaidi, L.; Fadhel, M.A.; Oleiwi, S.R.; Al-Shamma, O.; Zhang, J. DFU_QUTNet: Diabetic foot ulcer classification using novel deep convolutional neural network. *Multimed. Tools Appl.* **2019**, 1–23. [CrossRef]
10. LeCun, Y.; Bengio, Y.; Hinton, G. Deep learning. *Nature* **2015**, *521*, 436. [CrossRef]
11. Patil, P.R.; Sable, G.S.; Anandgaonkar, G. Counting of WBCs and RBCs from blood images using gray thresholding. *Int. J. Res. Eng. Technol.* **2014**, *3*, 391–395.

12. Alomari, Y.M.; Abdullah, S.; Huda, S.N.; Zaharatul Azma, R.; Omar, K. Automatic detection and quantification of WBCs and RBCs using iterative structured circle detection algorithm. *Comput. Math. Methods Med.* **2014**, *2014*, 979302. [CrossRef]
13. Bhagavathi, S.L.; Niba, S.T. An automatic system for detecting and counting RBC and WBC using fuzzy logic. *Arpn J. Eng. Appl. Sci.* **2016**, *11*, 6891–6894.
14. Maitra, M.; Gupta, R.K.; Mukherjee, M. Detection and counting of red blood cells in blood cell images using Hough transform. *Int. J. Comput. Appl.* **2012**, *53*, 16. [CrossRef]
15. Thejashwini, M.; Padma, M.C. Counting of RBC's and WBC's Using Image Processing Technique. *Int. J. Recent Innov. Trends Comput. Commun.* **2015**, *3*, 2948–2953.
16. Mazalan, S.M.; Mahmood, N.H.; Razak, M.A.A. Automated red blood cells counting in peripheral blood smear image using circular Hough transform. In Proceedings of the 2013 1st IEEE International Conference on Artificial Intelligence, Modelling and Simulation, Kota Kinabalu, Malaysia, 3–5 December 2013; pp. 320–324.
17. Tulsani, H.; Saxena, S.; Yadav, N. Segmentation using morphological watershed transformation for counting blood cells. *IJCAIT* **2013**, *2*, 28–36.
18. Sreekumar, A.; Bhattacharya, A. Identification of sickle cells from microscopic blood smear image using image processing. *Int. J. Emerg. Trends Sci. Technol.* **2014**, *1*, 783–787.
19. Chintawar, I.A.; Aishvarya, M.; Kuhikar, C. Detection of sickle cells using image processing. *Int. J. Sci. Technol. Eng.* **2016**, *2*, 335–339.
20. Patil, D.N.; Khot, U.P. Image processing based abnormal blood cells detection. *Int. J. Tech. Res. Appl.* **2015**, *31*, 37–43.
21. Sahu, M.; Biswas, A.K.; Uma, K. Detection of Sickle Cell Anemia in Red Blood Cell. *Int. J. Eng. Appl. Sci. (Ijeas)* **2015**, *2*, 3.
22. Rexcy, S.M.A.; Akshaya, V.S.; Swetha, K.S. Effective use of image processing techniques for the detection of sickle cell anemia and presence of plasmodium parasites. *Int. J. Adv. Res. Innov. Ideas Educ.* **2016**, *2*, 701–706.
23. Rakshit, P.; Bhowmik, K. Detection of abnormal findings in human RBC in diagnosing sickle cell anaemia using image processing. *Procedia Technol.* **2013**, *10*, 28–36. [CrossRef]
24. Revathi, T.; Jeevitha, S. Efficient watershed based red blood cell segmentation from digit al images in sickle cell disease. *Int. J. Sci. Eng. Appl. Sci.* **2016**, *2*, 300–317.
25. Bala, S.; Doegar, A. Automatic detection of sickle cell in red blood cell using watershed segmentation. *Int. J. Adv. Res. Comput. Commun. Eng.* **2015**, *4*, 488–491.
26. Gonzalez-Hidalgo, M.; Guerrero-Pena, F.A.; Herold-Garcia, S.; Jaume-i-Capó, A.; Marrero-Fernández, P.D. Red blood cell cluster separation from digital images for use in sickle cell disease. *IEEE J. Biomed. Health Inform.* **2014**, *19*, 1514–1525. [CrossRef] [PubMed]
27. Parvathy, H.B.; Hariharan, S.; Aruna, S.N. A Real Time System for the Analysis of Sickle Cell Anemia Blood Smear Images Using Image Processing. *Int. J. Innov. Res. Sci. Eng. Technol.* **2016**, *5*, 6200–6207.
28. Veluchamy, M.; Perumal, K.; Ponuchamy, T. Feature extraction and classification of blood cells using artificial neural network. *Am. J. Appl. Sci.* **2012**, *9*, 615.
29. Poomcokrak, J.; Neatpisarnvanit, C. Red blood cells extraction and counting. In Proceedings of the 3rd International Symposium on Biomedical Engineering, Changsha, China, 8–10 June 2008; pp. 199–203.
30. AbdulraheemFadhel, M.; Humaidi, A.J.; RazzaqOleiwi, S. Image processing-based diagnosis of sickle cell anemia in erythrocytes. In Proceedings of the 2017 Annual Conference on New Trends in Information & Communications Technology Applications (NTICT) IEEE, Baghdad, Iraq, 7–9 March 2017; pp. 203–207.
31. Sharma, V.; Rathore, A.; Vyas, G. Detection of sickle cell anaemia and thalassaemia causing abnormalities in thin smear of human blood sample using image processing. In Proceedings of the 2016 International Conference on Inventive Computation Technologies (ICICT), Coimbatore, India, 26–27 August 2016; pp. 1–5.
32. Rodrigues, L.F.; Naldi, M.C.; Mari, J.F. Morphological analysis and classification of erythrocytes in microscopy images. In Proceedings of the 2016 Workshop de Visão Computacional, Campo Grande, Brazil, 9–11 November 2016; pp. 1–6.
33. Chen, H.M.; Tsao, Y.T.; Tsai, S.N. Automatic image segmentation and classification based on direction texton technique for hemolytic anemia in thin blood smears. *Mach. Vis. Appl.* **2014**, *25*, 501–510. [CrossRef]
34. Acharya, V.; Kumar, P. Identification and red blood cell classification using computer aided system to diagnose blood disorders. In Proceedings of the 2017 IEEE International Conference on Advances in Computing, Communications and Informatics (ICACCI), Udupi, India, 13–16 September 2017; pp. 2098–2104.

35. Elsalamony, H.A. Anaemia cells detection based on shape signature using neural networks. *Measurement* **2017**, *104*, 50–59. [CrossRef]
36. Albayrak, B.; Darici, M.B.; Kiraci, F.; Öğrenci, A.S.; Özmen, A.; Ertez, K. Orak Hücreli Anemi Tespiti Sickle Cell Anemia Detection. In Proceedings of the 2018 IEEE Medical Technologies National Congress (TIPTEKNO), Magusa, Cyprus, 8–10 November 2018; pp. 1–4.
37. Chy, T.S.; Rahaman, M.A. Automatic Sickle Cell Anemia Detection Using Image Processing Technique. In Proceedings of the 2018 IEEE International Conference on Advancement in Electrical and Electronic Engineering (ICAEEE), Gazipur, Bangladesh, 22–24 November 2018; pp. 1–4.
38. Chy, T.S.; Rahaman, M.A. A Comparative Analysis by KNN, SVM & ELM Classification to Detect Sickle Cell Anemia. In Proceedings of the 2019 IEEE International Conference on Robotics, Electrical and Signal Processing Techniques (ICREST), Dhaka, Bangladesh, 10–12 January 2019; pp. 455–459.
39. Alzubaidi, L.; Al-Shamma, O.; Fadhel, M.A.; Farhan, L.; Zhang, J. Classification of Red Blood Cells in Sickle Cell Anemia Using Deep Convolutional Neural Network. In *International Conference on Intelligent Systems Design and Applications*; Springe: Cham, Switzerland, 2018; pp. 550–559.
40. Xu, M.; Papageorgiou, D.P.; Abidi, S.Z.; Dao, M.; Zhao, H.; Karniadakis, G.E. A deep convolutional neural network for classification of red blood cells in sickle cell anemia. *PLoS Comput. Biol.* **2017**, *13*, e1005746. [CrossRef]
41. Parthasarathy, D. WBC-Classification. Available online: https://github.com/dhruvp/wbc-classification/tree/master/Original_Images (accessed on 15 November 2019).
42. Wadsworth-Center. White Blood Cell Images. Available online: https://www.wadsworth.org/ (accessed on 10 November 2019).
43. Al-Dulaimi, K.; Chandran, V.; Banks, J.; Tomeo-Reyes, I.; Nguyen, K. Classification of white blood cells using bispectral invariant features of nuclei shape. In Proceedings of the 2018 IEEE Digital Image Computing: Techniques and Applications (DICTA), Canberra, Australia, 10–13 December 2018; pp. 1–8.
44. Labati, R.D.; Piuri, V.; Scotti, F. All-IDB: The acute lymphoblastic leukemia image database for image processing. In Proceedings of the 2011 18th IEEE International Conference on Image Processing, Brussels, Belgium, 11–14 September 2011; pp. 2045–2048.
45. Sickle Cells Anemia. Available online: http://sicklecellanaemia.org/ (accessed on 1 September 2019).
46. Fang, B.; Lu, Y.; Zhou, Z.; Li, Z.; Yan, Y.; Yang, L.; Jiao, G.; Li, G. Classification of Genetically Identical Left and Right Irises Using a Convolutional Neural Network. *Electronics* **2019**, *8*, 1109. [CrossRef]
47. Lee, H.; Lee, J. A Deep Learning-Based Scatter Correction of Simulated X-ray Images. *Electronics* **2019**, *8*, 944. [CrossRef]
48. Li, Y.; Richtarik, P.; Ding, L.; Gao, X. On the decision boundary of deep neural networks. *arXiv* **2018**, arXiv:abs/1808.05385. Available online: https://arxiv.org/abs/1808.05385 (accessed on 9 December 2019).
49. Kermany, D.S.; Goldbaum, M.; Cai, W.; Valentim, C.C.; Liang, H.; Baxter, S.L.; Dong, J. Identifying medical diagnoses and treatable diseases by image-based deep learning. *Cell* **2018**, *172*, 1122–1131. [CrossRef] [PubMed]
50. Yosinski, J.; Clune, J.; Bengio, Y.; Lipson, H. How transferable are features in deep neural networks? In Proceedings of the Neural Information Processing Systems 2014, Montreal, QC, Canada, 8–13 December 2014; pp. 3320–3328.
51. Pan, S.J.; Yang, Q. A survey on transfer learning. *IEEE Trans. Knowl. Data Eng.* **2009**, *22*, 1345–1359. [CrossRef]
52. Deng, J.; Dong, W.; Socher, R.; Li, L.J.; Li, K.; Fei-Fei, L. ImageNet: A large-scale hierarchical image database. In Proceedings of the 2009 IEEE conference on computer vision and pattern recognition, Miami, FL, USA, 20–25 June 2009; pp. 248–255.
53. Cook, D.; Feuz, K.D.; Krishnan, N.C. Transfer learning for activity recognition: A survey. *Knowl. Inf. Syst.* **2013**, *36*, 537–556. [CrossRef]
54. Cao, X.; Wang, Z.; Yan, P.; Li, X. Transfer learning for pedestrian detection. *Neurocomputing* **2013**, *100*, 51–57. [CrossRef]
55. Raghu, M.; Zhang, C.; Kleinberg, J.; Bengio, S. Transfusion: Understanding transfer learning for medical imaging. In Proceedings of the Neural Information Processing Systems 2019, Vancouver, BC, Canada, 8–14 December 2019; pp. 3342–3352.
56. Tan, C.; Sun, F.; Kong, T.; Zhang, W.; Yang, C.; Liu, C. A survey on deep transfer learning. In *International Conference on Artificial Neural Networks*; Springer: Cham, Switzerland, October 2018; pp. 270–279.

57. Van Dyk, D.A.; Meng, X.L. The art of data augmentation. *J. Comput. Graph. Stat.* **2001**, *10*, 1–50. [CrossRef]
58. Lv, E.; Wang, X.; Cheng, Y.; Yu, Q. Deep ensemble network based on multi-path fusion. *Artif. Intell. Rev.* **2019**, *52*, 151–168. [CrossRef]
59. Lv, E.; Wang, X.; Cheng, Y.; Yu, Q. Deep convolutional network based on pyramid architecture. *IEEE Access* **2018**, *6*, 43125–43135. [CrossRef]
60. Wang, J.; Wei, Z.; Zhang, T.; Zeng, W. Deeply-fused nets. *arXiv* **2016**, arXiv:Abs/1605.07716.
61. Simonyan, K.; Zisserman, A. Very deep convolutional networks for large-scale image recognition. *arXiv* **2014**, arXiv:Abs/1409.1556. Available online: https://arxiv.org/abs/1409.1556 (accessed on 9 December 2019).
62. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. ImageNet classification with deep convolutional neural networks. In Proceedings of the Neural Information Processing Systems 2012, Lake Tahoe, NV, USA, 3–6 December 2012; pp. 1097–1105.
63. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.
64. Srivastava, N.; Hinton, G.; Krizhevsky, A.; Sutskever, I.; Salakhutdinov, R. Dropout: A simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.* **2014**, *15*, 1929–1958.
65. Gual-Arnau, X.; Herold-García, S.; Simó, A. Erythrocyte shape classification using integral-geometry-based methods. *Med. Biol. Eng. Comput.* **2015**, *53*, 623–633. [CrossRef] [PubMed]
66. De Faria, L.C.; Rodrigues, L.F.; Mari, J.F. Cell classification using handcrafted features and bag of visual words. In Proceedings of the Workshop de Visão Computacional, Ilhéus-BA, Brazil, 12–14 November 2018.