Available online at www.sciencedirect.com

ScienceDirect

Cognitive Systems Research 52 (2018) 1036–1044

Cognitive Systems RESEARCH

www.elsevier.com/locate/cogsys

Detection of subtype blood cells using deep learning ☆

Prayag Tiwari [a,1], Jia Qian [b,1], Qiuchi Li [a], Benyou Wang [a], Deepak Gupta [c], Ashish Khanna [c], Joel J.P.C. Rodrigues [d,e,*], C. Victor Hugo [f]

[a] Department of Information Engineering, University of Padova, Italy
[b] Department of Applied Mathematics and Computer Science, Denmark Technology University, Denmark
[c] Maharaja Agrasen Institute of Technology, Delhi, India
[d] National Institute of Telecommunications (Inatel), Santa Rita do Sapucaí, MG, Brazil
[e] Instituto de Telecomunicações, Portugal
[f] de Albuquerque, University of Fortaleza (UNIFOR), FortalezalCE, Brazil

Received 15 July 2018; received in revised form 13 August 2018; accepted 25 August 2018
Available online 18 September 2018

Abstract

Deep Learning has already shown power in many application fields, and is accepted by more and more people as a better approach than the traditional machine learning models. In particular, the implementation of deep learning algorithms, especially Convolutional Neural Networks (CNN), brings huge benefits to the medical field, where a huge number of images are to be processed and analyzed. This paper aims to develop a deep learning model to address the blood cell classification problem, which is one of the most challenging problems in blood diagnosis. A CNN-based framework is built to automatically classify the blood cell images into subtypes of the cells. Experiments are conducted on a dataset of 13k images of blood cells with their subtypes, and the results show that our proposed model provide better results in terms of evaluation parameters.

Keywords: Blood cells; Classification; CNN

1. Introduction

The white blood cell (WBC), also called leukocytes, is a cellular component of the blood with a nucleus and without a hemoglobin. As an essential part of the immune system, it moves from blood to tissue and provide defense for fighting against the invasion of the foreign microorganisms, e.g., bacteria, viruses, and germs, by ingesting them, destroying infectious agents or by producing antibodies. The leukocyte can be categorized into five genres: *Eosinophils, lymphocytes, Neutrophils, Monocytes* and *Basophils*. The Neutrophils are the most abundant, and they are responsible for defending the bacteria or fungal infection. Eosinophils occupy around 2–4% of WBC, and act in response to allergies and parasite infection. Lymphocytes undertakes the task of the specific recognition of foreign agents and the consequent removal from the host. Monocytes are effective in direct destruction of pathogens and

☆ Fully documented templates are available in the elsarticle package on CTAN.

* Corresponding author at: National Institute of Telecommunications (Inatel), Santa Rita do Sapucaí, MG, Brazil.
E-mail addresses: prayag.tiwari@unipd.it (P. Tiwari), qiuchi.li@studenti.unipd.it (Q. Li), deepakgupta@mait.ac.in (D. Gupta), ashishkhanna@mait.ac.in (A. Khanna), joeljr@ieee.org (J.J.P.C. Rodrigues).

[1] Authors contributed equally to this work.

https://doi.org/10.1016/j.cogsys.2018.08.022
1389-0417/

cleanup of the debris from the infection sites. The counter of different white blood cells plays a significant role in the clinical diagnosis and test: it is an indicator that reflects the hidden infection within the body and alerts the hematologists as a signal, i.e., the abnormal increase in WBC is the so-called leukocytosis. It also helps doctors monitor the effectiveness of chemotherapy or radiation treatment in people with cancer (Doan, 1954).

The detection and distinguishment of diverse WBC and the further counting of the corresponding proportion is critical due to the richness of clinical meaning behind it. It is painstaking and of low-efficiency if we manually differentiate the leukocytes under the microscopes, from which the automatic classification based on the images of WBC emerges (Sabino, da Fontoura Costa, Rizzatti, & Zago, 2004). Usually, the automatic classification approaches are present with several main steps: preprocessing, segmentation, feature extraction and classification. The preprocessing procedure primarily refers to the attempt of removing the noises or some artifacts from the images to output the contrast images. The segmentation can be considered as the operation of segmenting the WBC from the background of the smear images or extracting the region of interest (ROI). The consequent step is to build a representative feature vector for every type of WBC, and the classification will work based on it. In this very step, the hematologist sometimes may be involved to determine the features.

In this paper, we propose a CNN-based model which gives 94% accuracy for the 2 labels i.e. polynuclear and mononuclear, 78% accuracy for the 4 labels i.e. eosinophil, lymphocyte, neutrophil and monocyte. Our model can automatically classify the blood cell type in order to save time and enhance clinical efficiency. The rest of the paper is organized as follows: Section 1 describes the introduction, Section 2 describes about the used baseline models to compare with our proposed model, Section 3 provides the literature survey where past relevant work is introduced, Section 4 provides the proposed methodologies followed with architecture and algorithm, Section 5 provides the data records, Section 6 provides the experiment followed by used setups and result analysis with the discussion, and Section 7 provides the conclusion and possible future works.

2. Background

In this section, some details about Naive Bayes and SVM has been described because these methods are used as a baseline approach to compare with our proposed model.

2.1. Naive Bayes

Bayes theorem with the independence supposition between the predictors is the base of the Naive Bayes classifier. Bayes theorem presents us a way to compute the posterior probability, $P(y|x)$ from the $P(y), P(x)$ and $P(x|y)$.

Naive Bayes classifier infer that there is in-dependency among each predictor for the given class. This assumption is often known as conditional in-dependency of class and given equation provide better understanding,

$$P(y|x) = \frac{P(x|y)P(y)}{P(x)} \tag{1}$$

where, posterior probability of target for the given attribute or predictor is represented by $P(y|x)$, prior probability of the target is represented by $P(y)$, likelihood that the probability of attribute for given class is represented by $P(x|y)$ and prior probability of attribute or predictor is represented by $P(X)$.

2.2. Support vector machine

SVM is used for classification as well as for regression purpose also. The main aim of SVM for classification purpose is to build a hyperplane in a multi dimensional space that partition the cases of distinct class labels. For building a optimal hyperplane, SVM utilize a iterative algorithm during training session, which is utilized to diminish the error function. SVM is classified into 2 types with respect to the type of error function.

SVM 1: Minimization of error function persist during the training session and can be written as:

$$\frac{v^T v}{2} + Const \sum_{k=1}^{M} \theta_k \tag{2}$$

which subjects to the limitations:

$$y_k(v^T \varphi(x_k) + b) \geqslant 1 - \theta_k,\ \theta_k \geqslant 0,\ k = 1, 2, \ldots\ldots, M \tag{3}$$

where *Const* means capacity constant, b is constant as well, v is the vector coefficients, θ_k represents the parameter for supervising the non-separable input data, $y \in \{+1, -1\}$ describes the labels of the class, independent variable is represented by x_k. The kernel φ transform from the input data to the feature space. Larger the value of C so the error can be penalized more that is why c must be opted carefully which could be helpful to avoid over-fitting.

SVM 2: In contrast to the SVM 1, SVM 2 helps to diminish the error function:

$$\frac{v^T v}{2} - w\rho + \frac{1}{M} \sum_{k=1}^{M} \theta_k \tag{4}$$

which subjects to the limitations:

$$y_k(v^T \varphi(x_k) + b) \geqslant \rho - \theta_k,\ \theta_k \geqslant 0,\ k = 1, 2, \ldots\ldots, M,\ and,\ \rho \geqslant 0 \tag{5}$$

3. Literature survey

The identification of WBC has been studied and explored by many researchers and practitioners with the implementation of different methodologies. Image processing techniques are largely involved in this procedure before

applying classification algorithms, such as separating the region of interest from the background, locating the nucleus of WBC (called nucleoli) and even in the preprocessing, removing and cleaning the noises from the microscopic images. To some extent, the segmentation from the background is critical as claimed by Su, Cheng, and Wang (2014), and many approaches have come up to address it, such as clustering (Sheikh, Zhu, & Micheli-Tzanakou, 1996), thresholding (Bikhet, Darwish, Tolba, & Shaheen, 2000; Nilufar, Ray, & Zhang, 2008; Yampri, Pintavirooj, Daochai, & Teartulakarn, 2006), morphological operator (Di Rubeto, Dempster, Khan, & Jarra, 2000; Osowski, Siroic, Markiewicz, & Siwek, 2009), Gram-Schmidt orthogonalization method (Rezatofighi & Soltanian-Zadeh, 2011), edge detection (Cseke, 1992), region growing (Chassery & Garbay, 1984), watershed (Jiang, Liao, & Dai, 2003), colors (Salihah et al., 2010; Wu et al., 2006) and so on.

A review based on the segmentation techniques (Adollah et al., 2008) argues that the conventional color-based and the thresholding methods are simple at the sacrifice of the accuracy, whereas the methods like region growing may offer a high precision with a high-computation cost. Referring to the color-based segmentation methods, some methods work directly on the RGB color space, whilst others instead on HSI or CMYK color space. In general, the S-component-based methods outperform the RGB-based ones. Putzu et al. (2013) attempts to build the feature vector by leveraging the CMYK color models. They find out that all the other components except white blood cells have some amount of yellow color present in them, whereas leukocytes show a good contrastion in the Y component of CMYK color model.

Our method will bypass the stringent demand of constructing a good feature profile by studying and analyzing the pixels of the microscopic images directly. As the procedure of hand-engineered features could be tricky but also critical, the expertise in different domains is required. However, sometimes the suggestion from the domain-based specialists may also have their own biases or subjective preferences. Our method takes the pixels of images (of three channels, RGB) as the input, and an algorithm automatically learns and harvests the critical variables during the training. The variables may be morphological or related to the color, but the algorithm will take good care of them no matter what. This preferable attribute differentiates our method from the work mentioned above by the other researchers.

4. Principle of convolutional neural networks

4.1. Methodology

CNN is a category of Artificial Neural Network (ANN), which has been proven to be useful in image classification and object recognition. It takes the raw pixels as input and produces an outcome indicating the probabilities that the input belongs to different classes. CNN gains its high reputation by advocating an innovative architecture that may address a big hurdle existing in the conventional neural network: a massive amount of parameters to tune in the training stage. Instead of implementing the fully-connected structure in every layer, CNN imposes two additional layers, convolution and pooling, that may significantly reduce the magnitude of parameters. Convolution layer is entitled by the convolutional operation, with the purpose of extracting the features from the input images. Besides, the kernel (filter) size and stride need to be determined in advance. The kernel is a numeric matrix, on which an element-wise operation between it and the input image is carried out. In the example in Fig. 1, the green matrix represents the pixel of the original image, the yellow matrix is the kernel or filter, whereas the pink matrix is the convolved feature matrix. The pink matrix is computed by moving the red dash square across the green matrix with the pre-set stride. The modification of kernel generates the features that have the variant effects, e.g., identification of the object, the edge detection, sharpen the image, etc. The pooling procedure is also called sub-sampling or down-sampling, which is designed to reduce the convolved features produced from the convolution operator with the

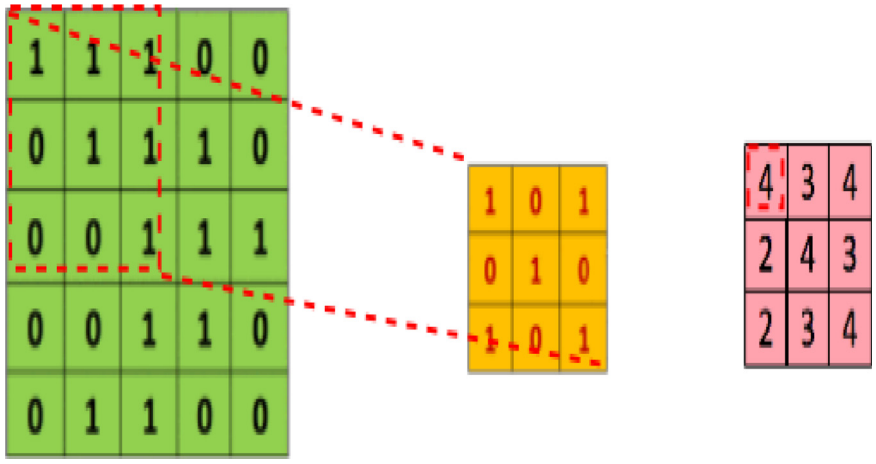

Fig. 1. An example of convolution operation: image pixel matrix, kernel and convolved feature (source).

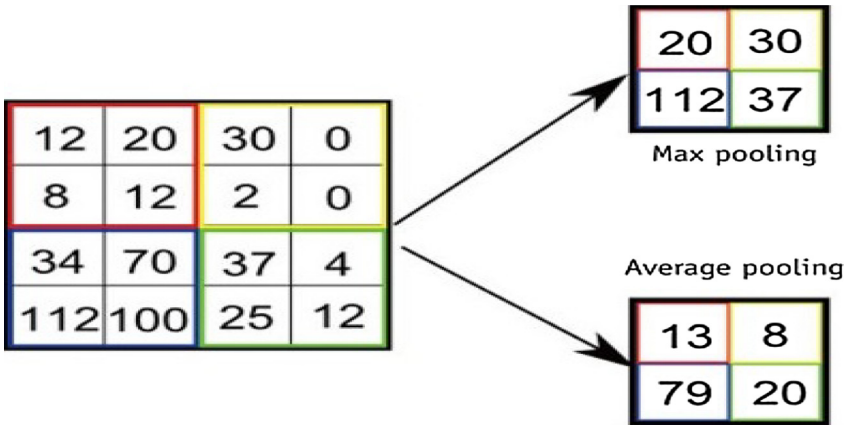

Fig. 2. An example of max pooling and average pooling operation (source).

incentive of remaining the considerable information. There are different pooling techniques, for instance, maximum, average, summation, etc. We use the maximize pooling in our method, an example is illustrated in Fig. 2. In a fully-connected layer, one node is connected to all the nodes in the previous layer. Besides, more than one hidden layer may apply, and the output layer may use the diverse classifying operator, e.g., softmax, rectified linear unit, etc.

4.2. Proposed architecture

The architecture of our proposed model is described in Fig. 3. It contains two convolutional layers and one pooling layer, and it is followed by the fully-connected structure with one hidden layer and output layer. All the detailed settings are listed as the follows:

1. Two convolution layers
   - First layer: Kernel size: 3×3, number of output filters: 32, initializer: Glorot uniform
   - Second layer: Kernel size: 3×3, number of output filters: 64, initializer: Glorot uniform
   - Activation function: ReLu
   - Stride: 1
   - Input image size: 60×80 (3 channels)
2. Pooling Layer
   - Pooling type: Maximize
   - Pool size: 2×2
   - Dropout: 0.25
3. Fully-connected Layer: Hidden Layer
   - Number of nodes:
   - Activation Function: ReLu
4. Fully-connected Layer: Output Layer
   - Number of nodes: number of classes
   - Activation Function: Softmax
5. Loss Function: Cross Entropy

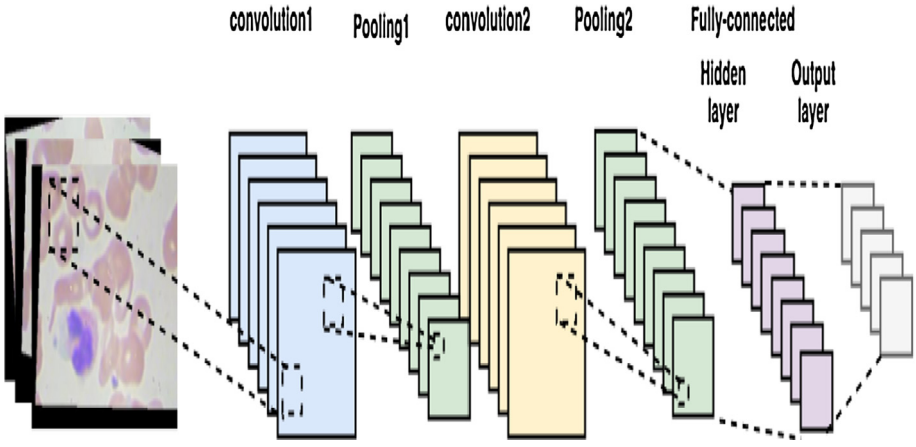

Fig. 3. Architecture of the Proposed Model.

4.3. Proposed algorithm

In proposed algorithm, forward pass and backward propagation algorithm has been implemented.

Forward Pass: Output ($P^{l,j}_{x,y}$) of the given neuron of j^{th} row, k column in the given l^{th} convolution layer with the j^{th} feature:

$$P^{l,j}_{x,y} = \tanh\left(\sum_{r=0}^{j_h}\sum_{t=0}^{f-1}\sum_{c=0}^{j_w} W^{j,t}_{r,c} P^{l-1,t}_{x+r,x+c} + bias^{l,j}\right) \tag{6}$$

where f is the No. of convolution cores in the pattern of feature. ($P^{l,j}_{x,y}$) neuron of the given row x and y column in the l^{th} subsample layer with the j^{th} feature:

$$P^{l,j}_{x,y} = \tanh\left(W^k \sum_{r=0}^{S_h}\sum_{c=0}^{S_w} P^{l-1,t}_{x_{S_h}+r,y_{S_w}+c} + bias^{l,j}\right) \tag{7}$$

so the output of the k_{th} neuron in the l_{th} hidden layer H will be:

$$P_{l,k} = \tanh\left(\sum_{x=0}^{S_h}\sum_{j=0}^{S-1}\sum_{y=0}^{S_w} W^{k,j}_{x,y} P^{l-1,t}_{x,y} + bias^{l,k}\right) \tag{8}$$

where number of pattern feature in simple layer is represented by s here so output of the i^{th} neuron and l^{th} output layer P

$$P_{l,i} = \tanh\left(\sum_{k=0}^{H} W^{l}_{i,k} P_{l-1,k} + bias^{l,i}\right) \tag{9}$$

Back Propagation: Output deviation of the j_{th} neuron in the output layer o can be written as $d(P^o_j) = y_j - t_j$ and the input deviation of the j_{th} neuron in the output layer is $d(I^o_j) = y_j - t_j\delta'(v_j) = \delta'(v_j d(P^o_j)$. Bias and weight variation of the j_{th} neuron in the output is $\Delta bias^o_j = d(I^o_j)$, $\Delta W^o_{j,x} = d(P^o_j)y_{j,x}$. In the hidden layer H, output bias of the j_{th} neuron is $d(P^H_j) = \sum_{i=0}^{i<a} d(I^o_i)W_{i,j}$ where a is the number of different pictures. In the hidden layer H, input bias of the j_{th} neuron is $d(I^H_j) = \delta'(v_j)d(P^H_j)$. Bias and weight variation in the column y and row x in the n^{th} feature pattern, previous layer in the head of j^{th} neurons in the hidden layer H is $\Delta bias^H_j = d(I^H_j)$, $\Delta W^{H,j}_{n,x,y} = d(I^H_j)y^n_{x,y}$. Output bias of column y and row x in the n^{th} feature pattern with S sub sample layer is $d(P^{S,n}_{x,y}) = \sum_k^b d(I^H_{n,x,y})W^{H,j}_{n,x,y}$ where b is the number of neurons in the hidden layer H. Input bias of column y and row x in the n^{th} feature pattern with S sub sample layer is $d(I^{S,n}_{x,y}) = \delta'(v_j)d(P^{S,n}_{x,y})$. Bias and weight variation in the column y and row x in the n^{th} feature pattern, S sub sample layer is $\Delta bias^{S,n} = \sum_{y=0}^{f_w}\sum_{x=0}^{f_h} d(P^{S,n}_{x,y})$, $\Delta W(S,n) = \sum_{y=0}^{f_w}\sum_{x=0}^{f_h} d(I^{S,n}_{x/2,y/2})P^{C,n}_{x,y}$. Convolution layer is represented by C here. So, Output bias of column y and row x in the j^{th} feature pattern with C convolution layer can be written as:

$$d(P^{C,j}_{x,y}) = d(I^{S,j}_{x/2,y/2})W_j \tag{10}$$

Input bias of column y and row x in the j^{th} feature pattern with C convolution layer can be written as:

$$d(I^{C,j}_{x,y}) = \delta'(v_j)d(P^{C,j}_{x,y}) \tag{11}$$

Weight variation of column c and row r in the n^{th} convolution core in addition with j^{th} feature pattern the given l^{th} layer with C convolution layer.

$$\Delta W^{j,n}_{r,c} = \sum_{y=0}^{f_w}\sum_{x=0}^{f_h} d(I^{C,j}_{x,y})P^{l-1,n}_{x+r,y+c} \tag{12}$$

So the total bias variation can be written of the convolution core

$$\Delta bias^{C,j} = \sum_{y=0}^{f_w}\sum_{x=0}^{f_h} d(I^{C,j}_{x,y}) \tag{13}$$

4.4. Evaluation parameters

There are some parameters often used in machine learning and deep learning in order to check the performance of the proposed model. Accuracy, Precision, Recall and F-measure has been used to evaluate the performance of the models and that is mentioned in the given equations. Confusion matrix can be seen from the given table

	Positive	Negative
Positive	TP (True Positive)	FP (False Positive)
Negative	FN (False Negative)	TN (True Negative)

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{14}$$

$$Precision = \frac{TP}{TP + FP} \tag{15}$$

$$Recall = \frac{TP}{TP + FN} \tag{16}$$

$$F1score = \frac{2.(Precision.Recall)}{(Precision + Recall)} \tag{17}$$

5. Data record

The experiment dataset contains around 13k augmented pictures of blood cells with having the blood cell type labels. There are around 3000 pictures for each blood cell

Table 1
Compare Double Convolution Layer Neural Network (DCLNN) with NB and SVM in terms of two classes and four classes.

Method	Class	Precision	Recall	F1 score
	Four classes			
NB	NEUTROPHIL	0.28	0.32	0.29
	EOSINOPHIL	0.28	0.26	0.27
	MONOCYTE	0.39	0.41	0.40
	LYMPHOCYTE	0.25	0.22	0.23
	Average	0.30	0.30	0.30
SVM	NEUTROPHIL	0.28	0.27	0.27
	EOSINOPHIL	0.27	0.31	0.29
	MONOCYTE	0.40	0.50	0.44
	LYMPHOCYTE	0.17	0.11	0.13
	Average	0.28	0.30	0.28
DCLNN	NEUTROPHIL	**0.6**	**0.94**	**0.74**
	EOSINOPHIL	**0.95**	**0.61**	**0.74**
	MONOCYTE	**0.97**	**0.77**	**0.86**
	LYMPHOCYTE	**1**	**1**	**1**
	Average	0.88	0.83	0.83
	Two classes			
NB	Mononuclear	0.56	0.51	0.54
	Polynuclear	0.56	0.61	0.58
	Average	0.56	0.56	0.56
SVM	Mononuclear	0.56	0.52	0.54
	Polynuclear	0.55	0.59	0.57
	Average	0.56	0.56	0.56
DCLNN	Mononuclear	**0.98**	**0.88**	**0.92**
	Polynuclear	**0.89**	**0.98**	**0.93**
	Average	0.93	0.93	0.93

types which are into 5 different categories with the type of cell i.e. eosinophil, basophil, lymphocyte, neutrophil and monocyte. Basophil is not taken into consideration during analysis because there was no image of basophil in the obtained dataset. There is original dataset accompanied with augmented one which has 400 pictures in jpeg file with having two blood cell labels i.e. polynuclear and mononuclear. More details about dataset can be found here https://bigpictureeducation.com/blood-cells-images.

6. Experiments

6.1. Setups

For this analysis, these setups has been utilized in order to achieve the result and can be seen below.

Processor	Intel(R) Core(TM) i7-7700K CPU @ 4.20 GHz
RAM	16.0 GB
Operating System	Ubuntu 18.04
Graphic Card	Nvidia GeForce GTX 1060 3 GB

6.2. Results analysis and discussion

In the first array of experiments, we validate our proposed double convolution layer neural networks (DCLNN) by comparing the yet commonly used classifiers, Naive Bayer (NB) and Support Vector Machine (SVM) in Table 1. In the four-classes classification task, no matter which classifier and which matrix (precision, recall, F1), DCLNN thoroughly outperforms NB and SVM, labeled bold. Besides, DCLNN ends up with the average precision equal to 0.88, more than double times higher than SVM and NB. The recall produced by DCLNN is 0.83, almost triple times better than NB and SVM, F1 score equal to 0.83, visualized as the blue bars in Figs. 4–6 individually for the above mentioned three matrices. Talking to the two-classes classification, NB and SVM have better performance than the four-classes classification, however, DCLNN still outraces both of them for the mononuclear and polynuclear classes with respect to the precision, recall and F1 score in Table 1. Figs. 4–6, where the red bars indicate the binary-classes comparison.

Moreover, We conduct another series of experiments, with the attempt to determine the technique for the pooling layer: MaxPooling and AveragePooling. For four classes,

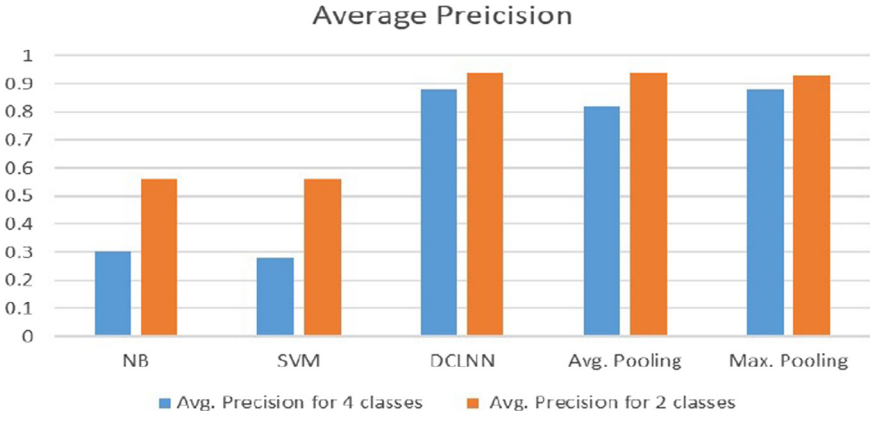

Fig. 4. Average Precision comparison among Naive Bayes, Support Vector Machine, Double Convolution Layer Neural Network (DCLNN), Average Pooling and Max Pooling based on two classes and four classes.

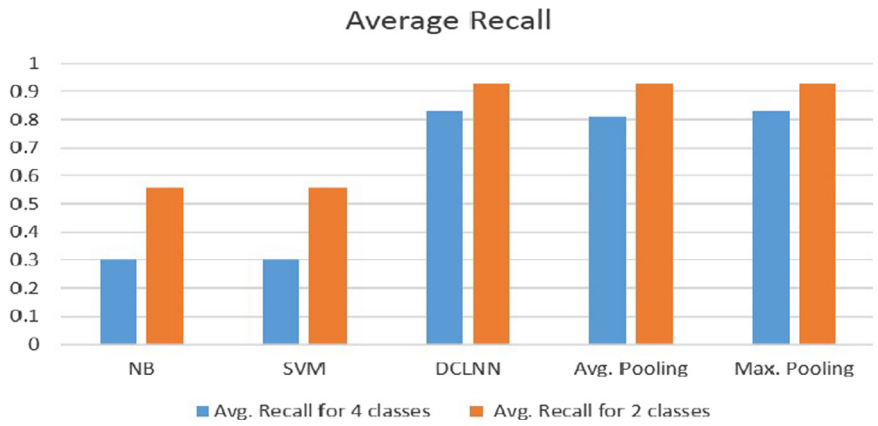

Fig. 5. Average Recall comparison among Naive Bayes, Support Vector Machine, Double Convolution Layer Neural Network (DCLNN), Average Pooling and Max Pooling based on two classes and four classes.

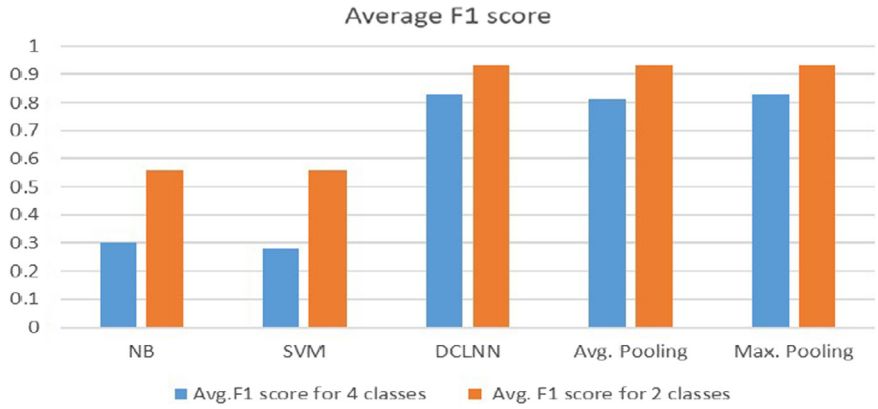

Fig. 6. Average F1 score comparison among Naive Bayes, Support Vector Machine, Double Convolution Layer Neural Network (DCLNN), Average Pooling and Max Pooling based on two classes and four classes.

Table 2
Compare MaxPooling and AveragePooling in terms of 2 classes and four classes.

Method	Class	Precision	Recall	F1 score
MaxPooling VS AveragePooling (four classes)				
AveragePooling	NEUTROPHIL	**0.64**	0.81	0.71
	EOSINOPHIL	0.81	**0.63**	0.71
	MONOCYTE	0.94	**0.8**	**0.86**
	LYMPHOCYTE	0.91	**1**	0.95
	Average	0.82	0.81	0.81
MaxPooling	NEUTROPHIL	0.60	**0.94**	**0.74**
	EOSINOPHIL	**0.95**	0.61	**0.74**
	MONOCYTE	**0.97**	0.77	**0.86**
	LYMPHOCYTE	**1**	**1**	**1**
	Average	0.88	0.83	0.83
MaxPooling VS AveragePooling (two classes)				
AveragePooling	Mononuclear	**0.99**	0.87	**0.93**
	Polynuclear	**0.89**	**0.99**	**0.94**
	Average	0.94	0.93	0.93
MaxPooling	Mononuclear	0.98	**0.88**	0.92
	Polynuclear	**0.89**	0.98	0.93
	Average	0.93	0.93	0.93

AveragePooling has better performance at some points, e.g., precision of neutrophil classifier, recall of the eosinophil classifier, etc. However, comprehensively MaxPooling outperforms AveragePooling shown in Table 2. The average performance(precision, recall and F1 score) is graphically demonstrated in Figs. 4–6. Referring to two classes, they are virtually identical.

Last but not least, to determine the optimal number of epochs for training and validation we exhaustively iterate variant number and compute the precision, in Fig. 7. The red line indicates the training precision and the blue line is the validation precision. We conclude that 25 is a good option trade-off between precision and computing time. Our proposed model outperform the traditional machine learning models which can be very useful in the medical field and this proposed model allow to get rid from the blood diagnosis problems.

7. Conclusion and future works

From the obtained experiments, Our proposed models suggest that implementation of deep learning enhance the classification task as compared to state-of-art models.

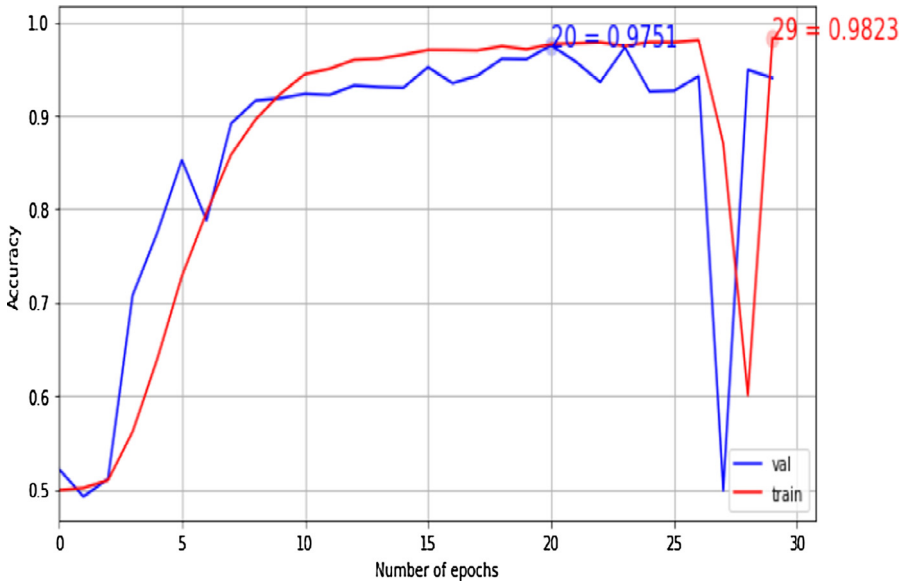

Fig. 7. Number of epochs with accuracy.

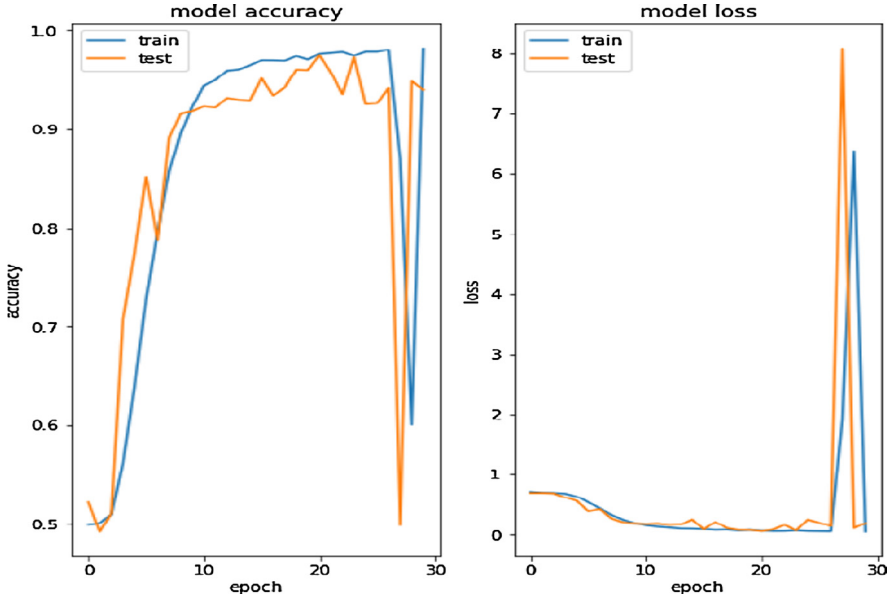

Fig. 8. Model accuracy and loss.

SVM and Naive Bayes has been utilized as a baseline to compare with the proposed CNN based model (DCLNN) and it compete in all aspects with the baseline approaches. Our proposed model can automatically classify the blood cell images into subtypes of the cells with high accuracy, precision and other evaluation parameters. This proposed model can be very beneficial for blood diagnosis in the medical field that can save a lot of time. We believe that there is always room for improvement in every field so as well in this field also. Researchers may implement this work on large dataset that may outperform the current results (see Fig. 8).

References

Adollah, R., Mashor, M., Nasir, N. M., Rosline, H., Mahsin, H., & Adilah, H. (2008). Blood cell image segmentation: A review. In *4th Kuala Lumpur international conference on biomedical engineering 2008* (pp. 141–144). Springer.

Bikhet, S. F., Darwish, A. M., Tolba, H. A., & Shaheen, S. I. (2000). Segmentation and classification of white blood cells. *ICASSP'00. Proceedings. 2000 IEEE international conference on acoustics, speech, and signal processing* (Vol. 4, pp. 2259–2261). IEEE.

Chassery, J.-M., & Garbay, C. (1984). An iterative segmentation method based on a contextual color and shape criterion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*(6), 794–800.

Cseke, I. (1992). A fast segmentation scheme for white blood cell images. *Conference C: Image, speech and signal analysis, proceedings., 11th IAPR international conference on pattern recognition* (Vol. III, pp. 530–533). IEEE.

Di Rubeto, C., Dempster, A., Khan, S., & Jarra, B. (2000). Segmentation of blood images using morphological operators. *Proceedings. 15th international conference on pattern recognition* (vol. 3, pp. 397–400). IEEE.

Doan, C. A. (1954). The white blood cells in health and disease. *Bulletin of the New York Academy of Medicine, 30*(6), 415.

Jiang, K., Liao, Q.-M., & Dai, S.-Y. (2003). A novel white blood cell segmentation scheme using scale-space filtering and watershed clustering. *2003 international conference on machine learning and cybernetics* (Vol. 5, pp. 2820–2825). IEEE.

Nilufar, S., Ray, N., & Zhang, H. (2008). Automatic blood cell classification based on joint histogram based feature and bhattacharya kernel. In *2008 42nd Asilomar conference on signals, systems and computers* (pp. 1915–1918). IEEE.

Osowski, S., Siroic, R., Markiewicz, T., & Siwek, K. (2009). Application of support vector machine and genetic algorithm for improved blood cell recognition. *IEEE Transactions on Instrumentation and Measurement, 58*(7), 2159–2168.

Putzu, L., & Di Ruberto, C. (2013). White blood cells identification and classification from leukemic blood image. In *International work-conference on bioinformatics and biomedical engineering* (pp. 99–106). Copicentro Editorial.

Rezatofighi, S. H., & Soltanian-Zadeh, H. (2011). Automatic recognition of five types of white blood cells in peripheral blood. *Computerized Medical Imaging and Graphics, 35*(4), 333–343.

Sabino, D. M. U., da Fontoura Costa, L., Rizzatti, E. G., & Zago, M. A. (2004). A texture approach to leukocyte recognition. *Real-Time Imaging, 10*(4), 205–216.

Salihah, A. A., Mashor, M., Harun, N. H., Abdullah, A. A., & Rosline, H. (2010). Improving colour image segmentation on acute myelogenous leukaemia images using contrast enhancement techniques. In *2010 IEEE EMBS conference on biomedical engineering and sciences (IECBES)* (pp. 246–251). IEEE.

Sheikh, H., Zhu, B., & Micheli-Tzanakou, E. (1996). Blood cell identification using neural networks. In *Proceedings of the 1996 IEEE twenty-second annual northeast on bioengineering conference* (pp. 119–120). IEEE.

Su, M.-C., Cheng, C.-Y., & Wang, P.-C. (2014). A neural-network-based approach to white blood cell classification. *The Scientific World Journal*.

Wu, J., Zeng, P., Zhou, Y., & Olivier, C. (2006). A novel color image segmentation method and its application to white blood cell image analysis. *2006 8th international conference on signal processing* (Vol. 2). IEEE.

Yampri, P., Pintavirooj, C., Daochai, S., & Teartulakarn, S. (2006). White blood cell classification based on the combination of eigen cell and parametric feature detection. In *2006 1ST IEEE conference on industrial electronics and applications* (pp. 1–4). IEEE.